Отчет по выполнению лабораторной работы №2

Дисциплина: Архитектура компьютера

Ефремова Полина Александровна

Содержание

# 1 Цель работы

Целью работы является применение средств контроля версий. А также очень важно 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 и репозитория курса на основе шаблона.
5. Создание репозитория курса на основе шаблона.
6. Настройка каталога курса.
7. Выполнение заданий для самостоятельной работы.

# 3 Теоретическое введение

Система контроля версий ( Version Control System, VCS ) — это инструмент, используемый разработчиками программного обеспечения для управления изменениями в исходном коде и других файловых ресурсах. Системы контроля версий разработаны специально для того, чтобы максимально упростить и упорядочить работу над проектом (вне зависимости от того, сколько человек в этом участвуют). СКВ дает возможность видеть, кто, когда и какие изменения вносил; позволяет формировать новые ветви проекта, объединять уже имеющиеся; настраивать контроль доступа к проекту ; осуществлять откат до предыдущих версий. 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. Демидова А. В. 14 Архитектура ЭВМ 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. Система контроля версий Git представляет собой набор программ командной строки. Доступ к ним можно получить из терминала посредством ввода команды git с различными опциями. Благодаря тому, что Git является распределённой системой контроля версий, резервную копию локального хранилища можно сделать простым копированием или архивацией.

Ниже показан перечень основных команд git (рис. 1)

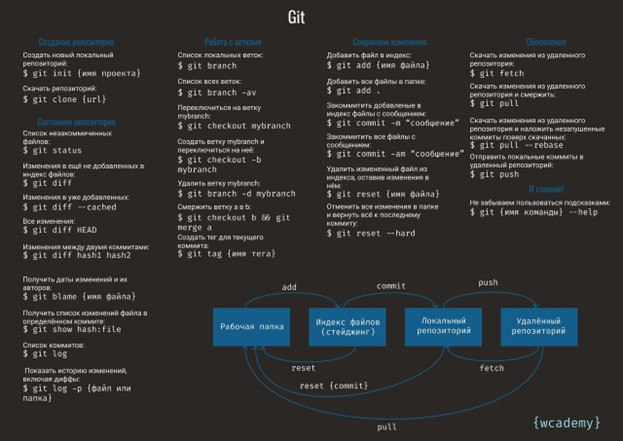


Рис. 1: Перечень основных команд git

# 4 Выполнение лабораторной работы

## 4.1 1.Настройка GitHub

Для выполнения лабораторной работы создаю учетную запись на https://github.com/ (рис. 2)

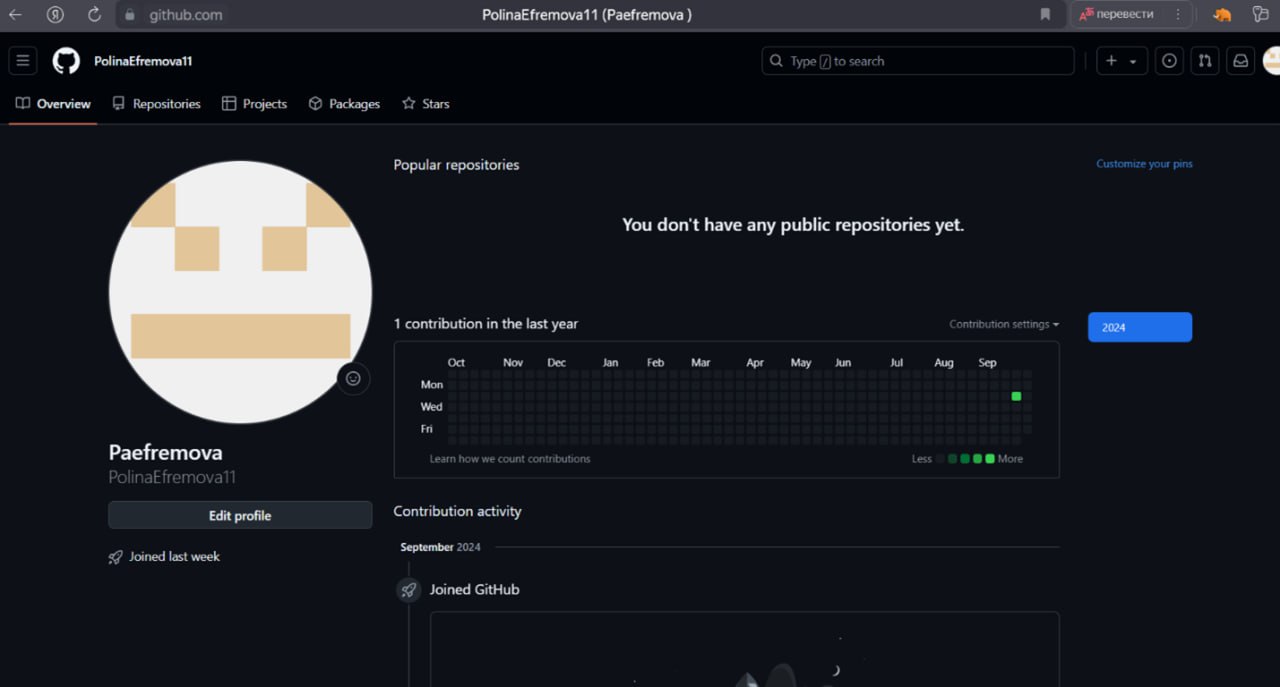


Рис. 2: Учетная запись

## 4.2 2.Базовая настройка git.

Делаю предварительную конфигурацию git. Захожу в терминал и ввожу команды, указывая свое имя и email (рис. 3)



Рис. 3: Ввод команд

Настраиваю utf-8 в выходе сообщений git. (рис. 4)

Настройка

Рис. 4: Настройка

Задаю имя начальной ветки, которую буду называть master (рис. 5)

Имя начальной ветки

Рис. 5: Имя начальной ветки

А также ввожу autocrlf и safecrlf (рис. 6)

Ввод команд

Рис. 6: Ввод команд

## 4.3 3.Создание SSH ключа

Для последующей идентификации пользователя на сервере репозиториев генерирую пару ключей (приватный и открытый). рис. 7)

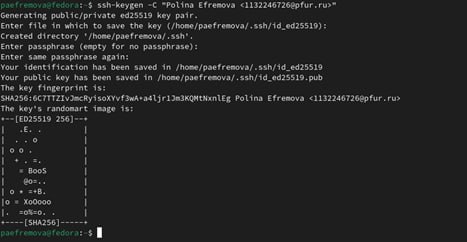


Рис. 7: Генерация ключей

Чтобы скопировать из локальной консоли ключ в буфер обмена, устанавливаю команду xclip (рис. 8)

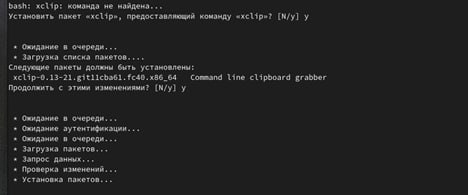


Рис. 8: Установка xclip

Теперь воспользуюсь командой xclip. (рис. 9)

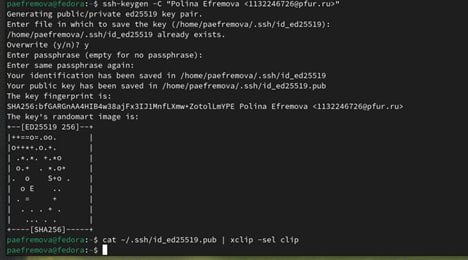


Рис. 9: Использование xclip

Вставляю ключ в появившееся на сайте поле, указываю его имя. (рис. 10)

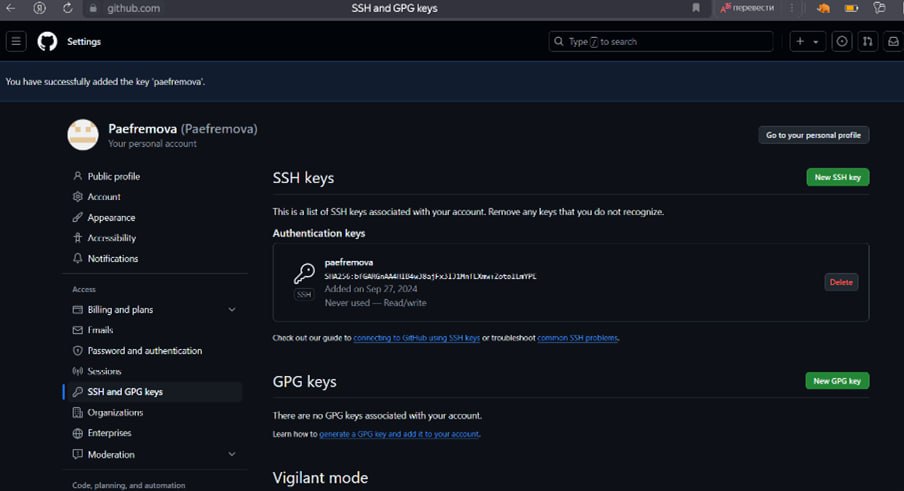


Рис. 10: Вставка ключа

## 4.4 4.Создание рабочего пространства, репозитория курса на основе шаблона.

Открываю терминал и создаю репозиторий для предмета «Архитектура компьютера». (рис. 11)

Создание репозитория

Рис. 11: Создание репозитория

## 4.5 5.Создание репозитория курса на основе шаблона.

Захожу на страницу репозитория с шаблоном курса, выбираю его в качестве своего нового. (рис. 12)

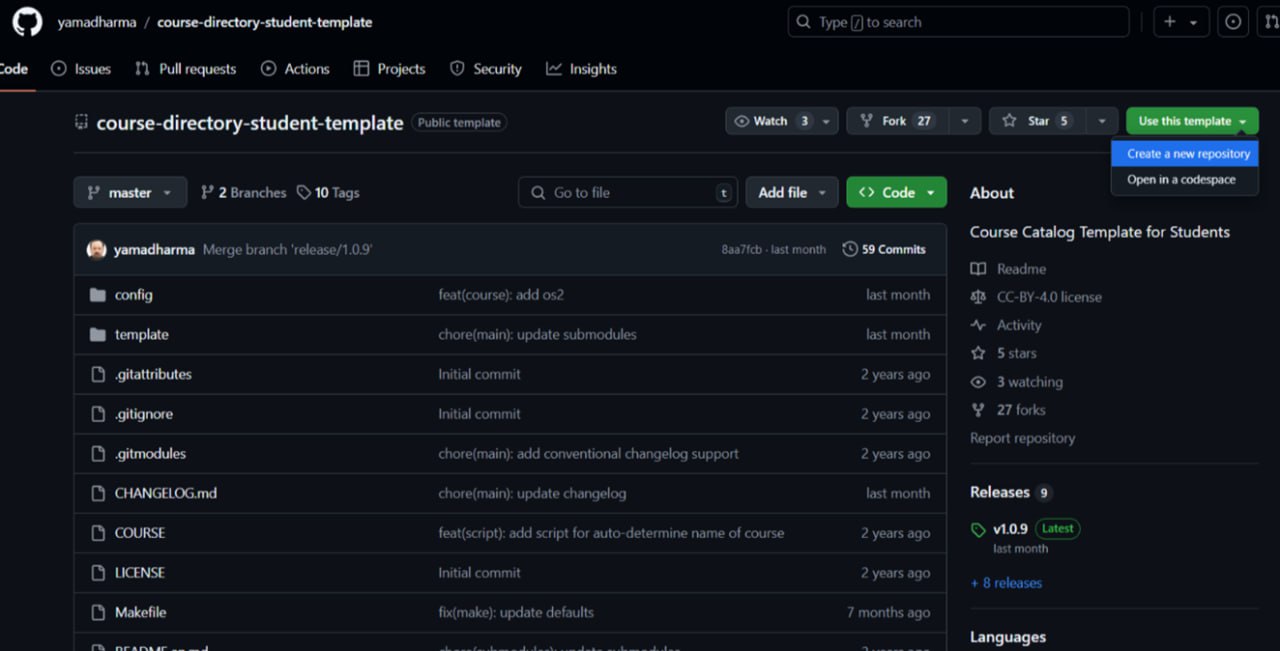


Рис. 12: Создание репозитория

Далее создаю его, задав ему имя. (рис. 13)

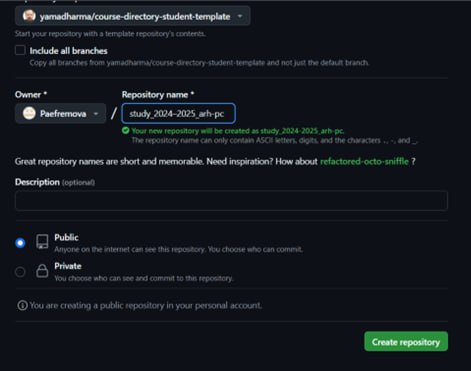


Рис. 13: Даю имя

Открываю терминал и перехожу в каталог курса. (рис. 14)

Переход в каталог курса

Рис. 14: Переход в каталог курса

Клонирую созданный репозиторий (рис. 15)

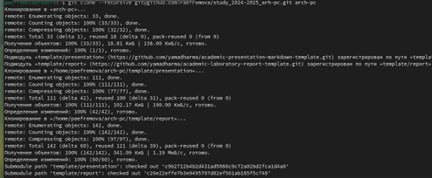


Рис. 15: Клонирование

## 4.6 6.Настройка каталога курса.

Перехожу в каталог курса и удаляю лишний файлы. (рис. 16)

Переход в каталог курса

Рис. 16: Переход в каталог курса

Создаю необходимые каталоги, отправляю файлы на сервер (рис. 17)

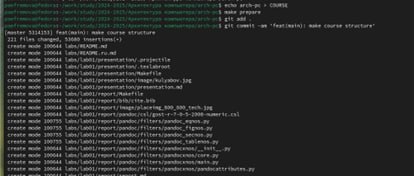


Рис. 17: Добавление файлов на сервер

В локальном репозитории проверяю результат выполненной работы (рис. 18)

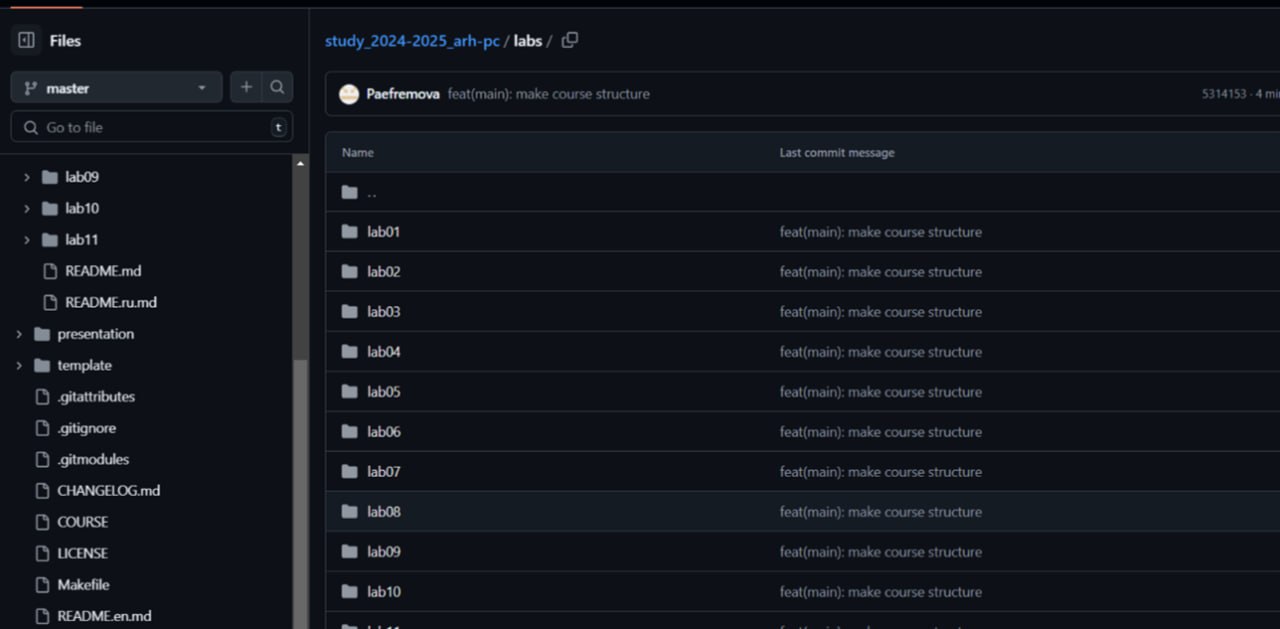


Рис. 18: Проверка высоплненной работы

# 5 7. Выполнение заданий для самостоятельной работы.

Создаю отчет по выполнению второй лабораторной работы в соответствующем каталоге. С помощью команды ls проверяю, создан ли файл.(рис. 19)

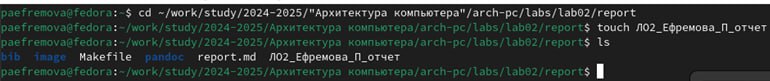


Рис. 19: Создание отчета по лабораторной работе

Для выполнения второго задания проверяю местонахождение своих лабораторных работ (рис. 20)

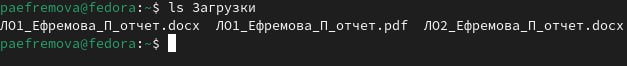


Рис. 20: Прверка местонахождения лабораторных работ

Копирую лабораторную работу с помощью утилиты cp, проверяю местонахождение файлов с помощью команды ls (рис. 21)

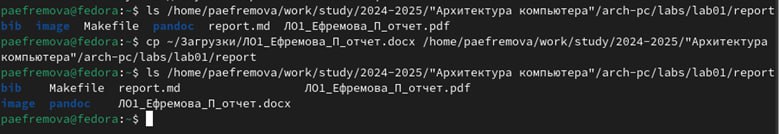


Рис. 21: Копирование

Для того чтобы загрузить эти файлы на GitHub, в первую очередь я использую команду git add. Так добавленные мной файлы станут отслеживаемыми. (рис. 22)

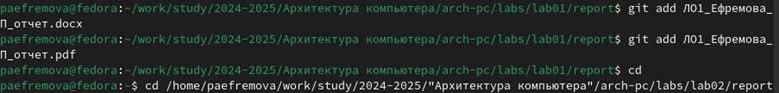


Рис. 22: Использование git add

Теперь осуществляю полноценный перенос файлов с помощью команды git commit -m “…” (рис. 23)

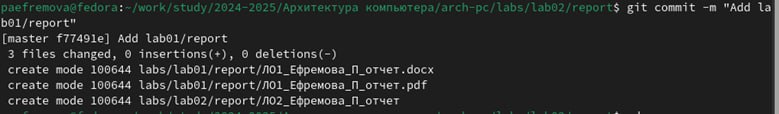


Рис. 23: Перенос файлов

Использую команды: git status и git push, чтобы опубликовать свои локальные коммиты. (рис. 24)

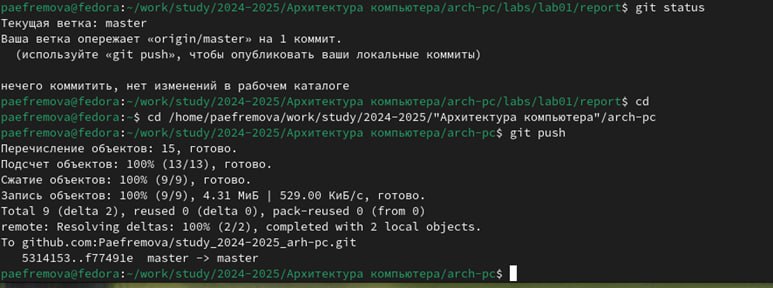


Рис. 24: Ввод команд

Перехожу в каталоги на GitHub, чтобы убедиться в том, что файлы находятся в нужных репозиториях. (рис. 25)

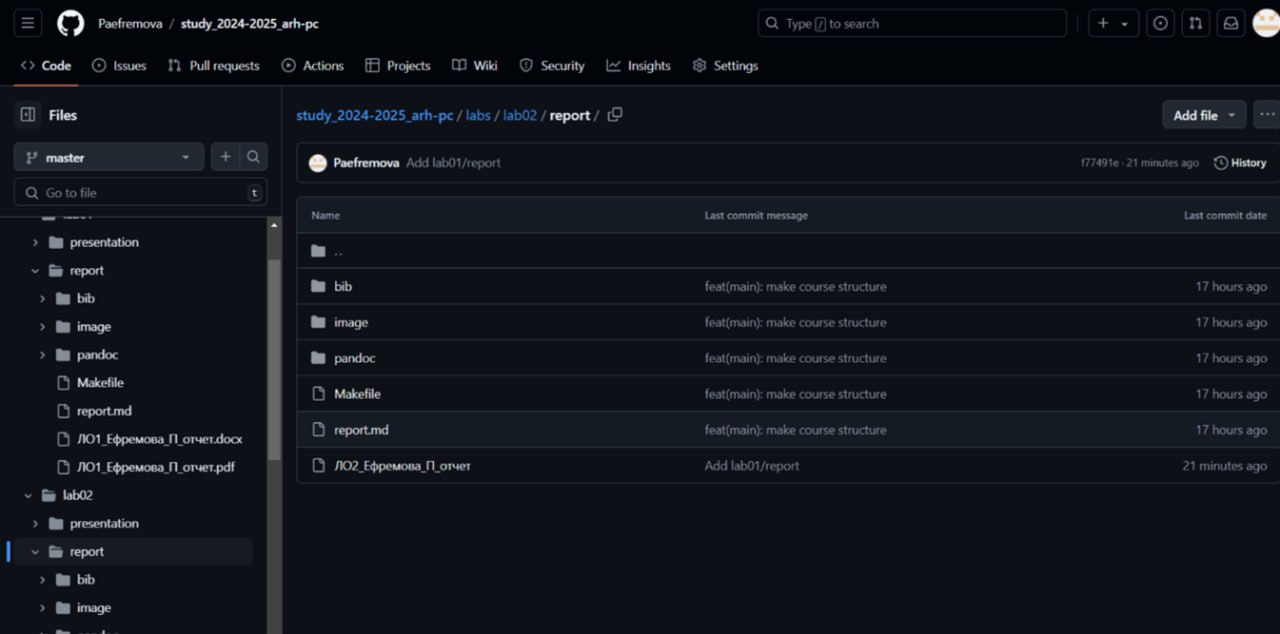


Рис. 25: Переход в каталоги

(рис. 26)

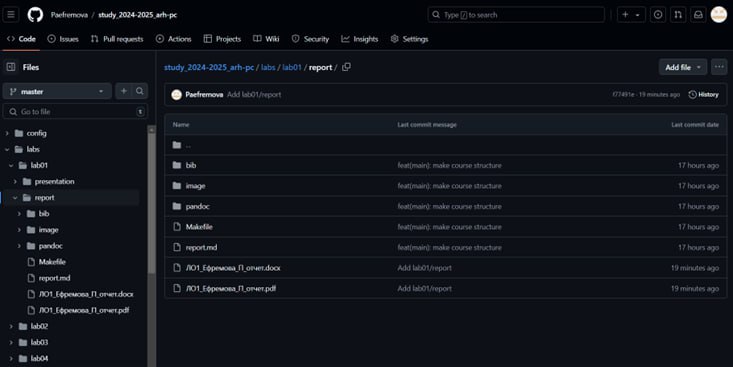


Рис. 26: Переход в каталоги

# 6 Выводы

В заключение хочется отметить, что данная лабораторная работа позволила мне научиться работать с системой Git. Я практиковала свои навыки в работе с командной строкой, теперь уже связывая выполнимое с директориями GitHub.

# 7 Список литературы

1.Архитектура ЭВМ https://esystem.rudn.ru/pluginfile.php/2089082/mod\_resource/content/0/Лабораторная%20работа%20№2.%20Система%20контроля%20версий%20Git.pdf

2.30 команд Git, необходимых для освоения интерфейса командной строки Git / Хабр https://habr.com/ru/companies/ruvds/articles/599929/

3.Система контроля версий: определение, функции, популярные решения https://gb.ru/blog/sistema-kontrolya-versij/