Отчёт по лабораторной работе №2

Дисциплина: архитектура компьютера

Арбатова Варвара Петровна

Содержание

# Цель работы

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

# Задание

1. Создать аккаунт в github
2. Базовая настройка github
3. Создать SSH-ключ
4. Настроить каталог курса на основе шаблона
5. Выполнить задания для самостоятельной работы

# 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.

# Выполнение лабораторной работы

2.4.1) Настройка github Регистрируюсь в github с корпоративной почтой и заполняю основные данные

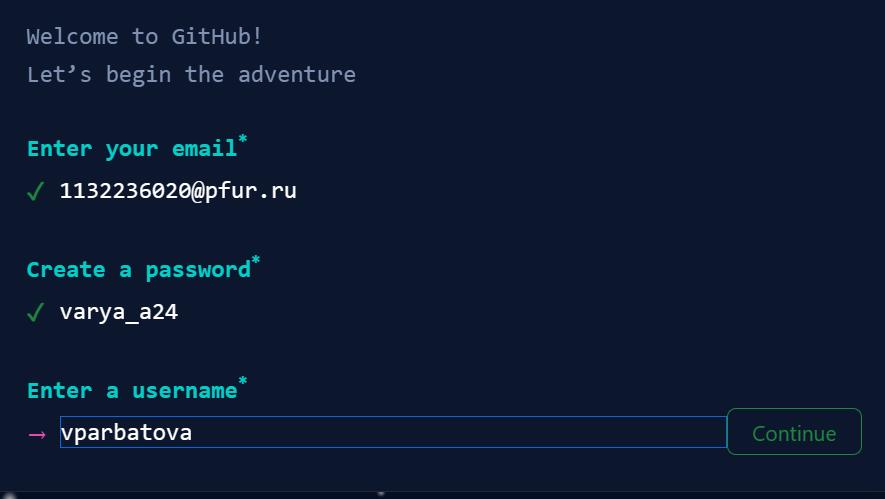


Figure 1: Рис1. Регистрация в github

2.4.2) Базовая настройка git Сначала делаю предварительную конфигурацию git. Настраиваю utf-8 в выводе сообщений git, задаю имя начальной ветки, параметры autocrlf и safecrlf

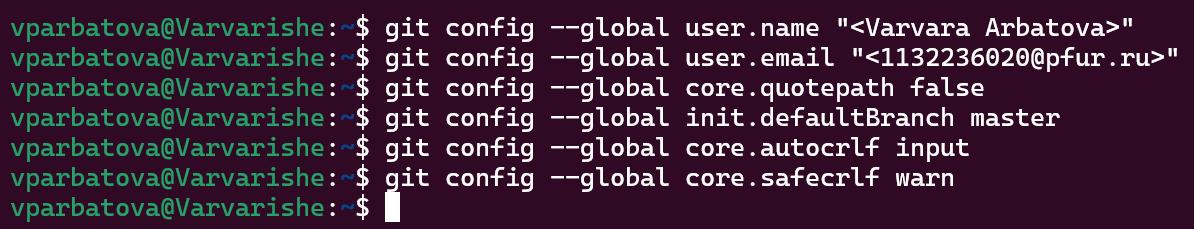


Figure 2: Рис2. Базовая настройка git

2.4.3) Создание SSH-ключа Генерирую пару ключей.

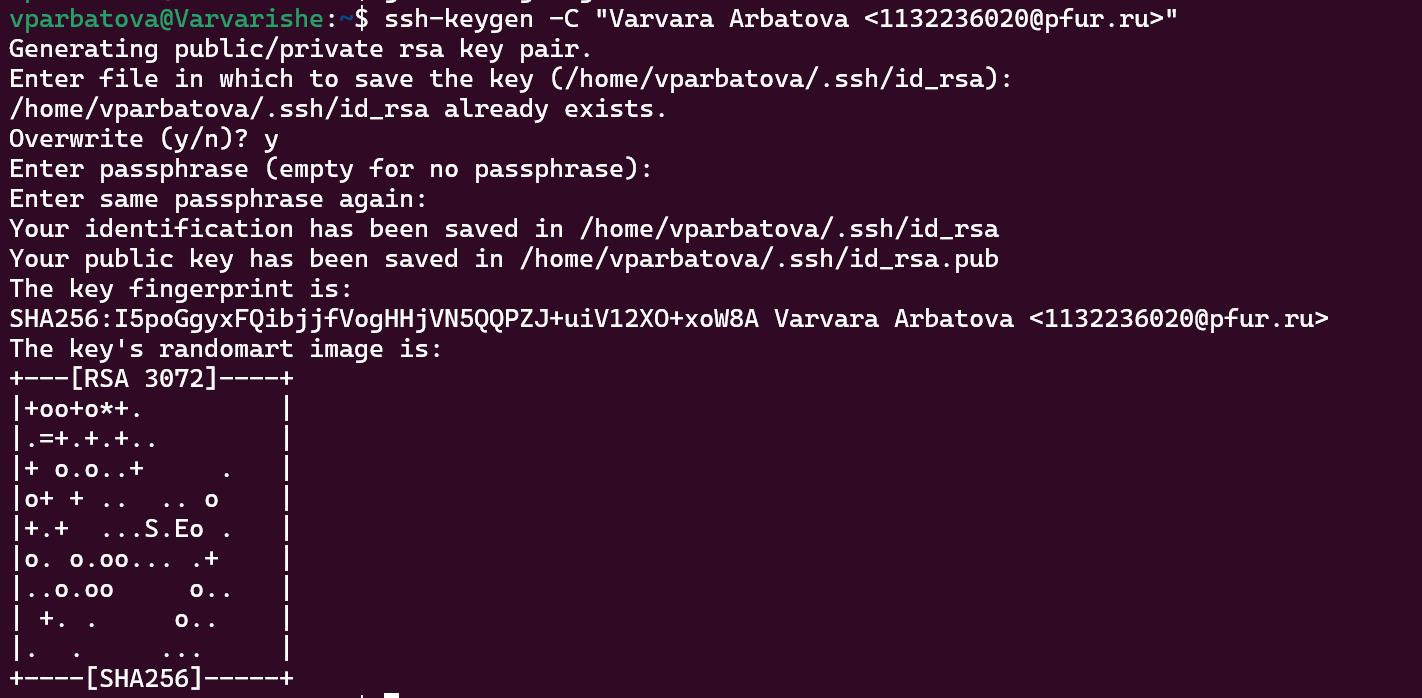


Figure 3: Рис3. Генерирую ключи

Перехожу в аккаунт на github

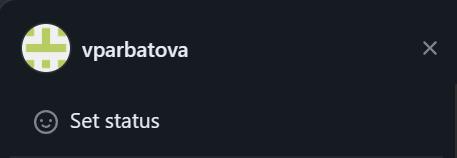


Figure 4: Рис4. Перехожу в свой аккаунт на github

Копирую из локальной консоли ключ в буфер обмена 

2.4.4) Сознание рабочего пространства и репозитория курса на основе шаблона С помощью команды mkdir -p создаю вложенные друг папки, проверяю корректность создания

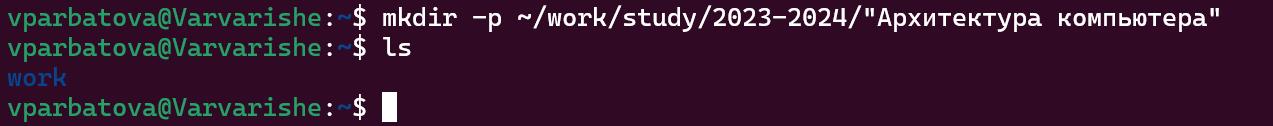


Figure 5: Рис7.

Создаю репозиторий на основе шаблона через web-интерфейс github.

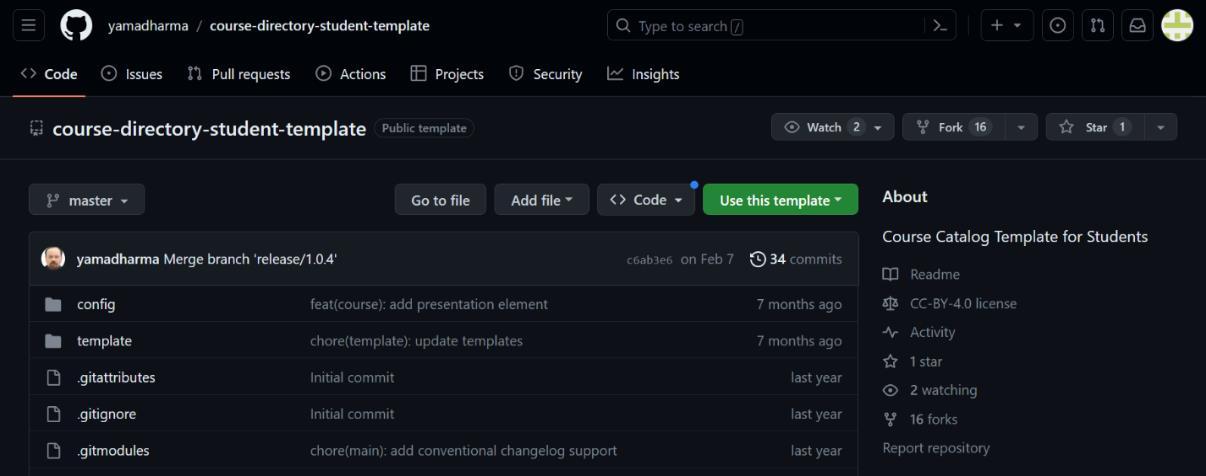


Figure 6: Рис8.

Репозиторий создан

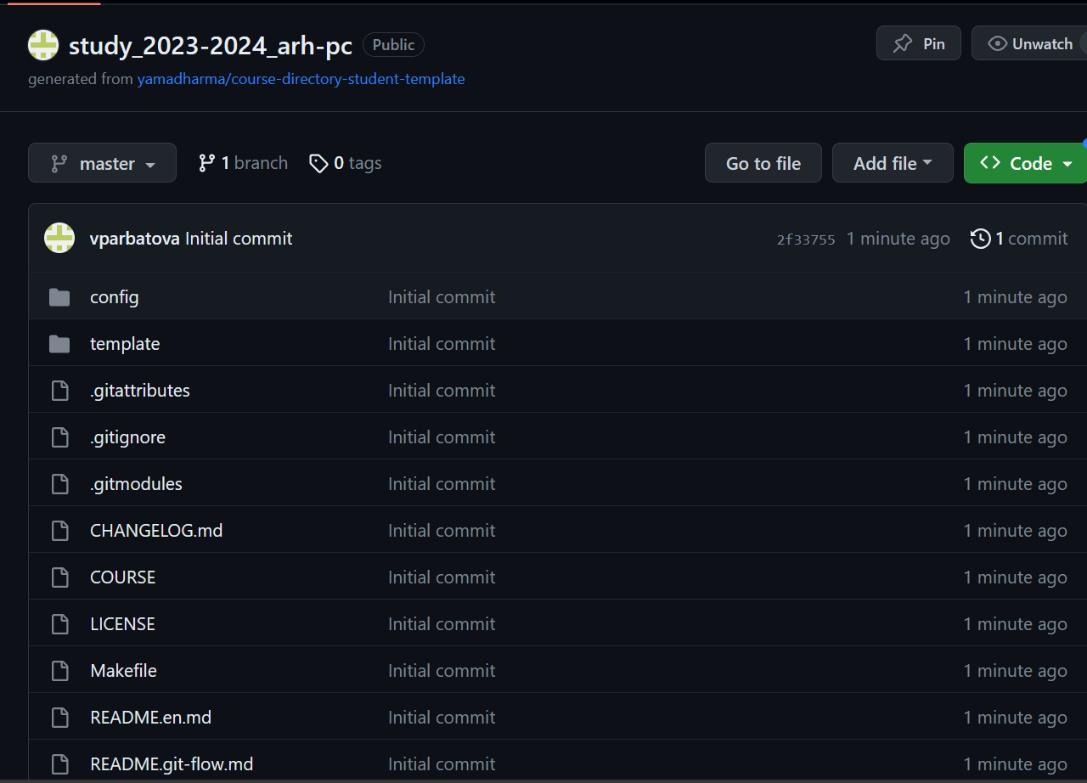


Figure 7: Рис9. Репозиторий создан

Перехожу в нужный мне каталог с помощью cd



Figure 8: Рис10. Переход в домашний каталог

Клонирую созданный репозиторий с помощью команды git clone –recursive

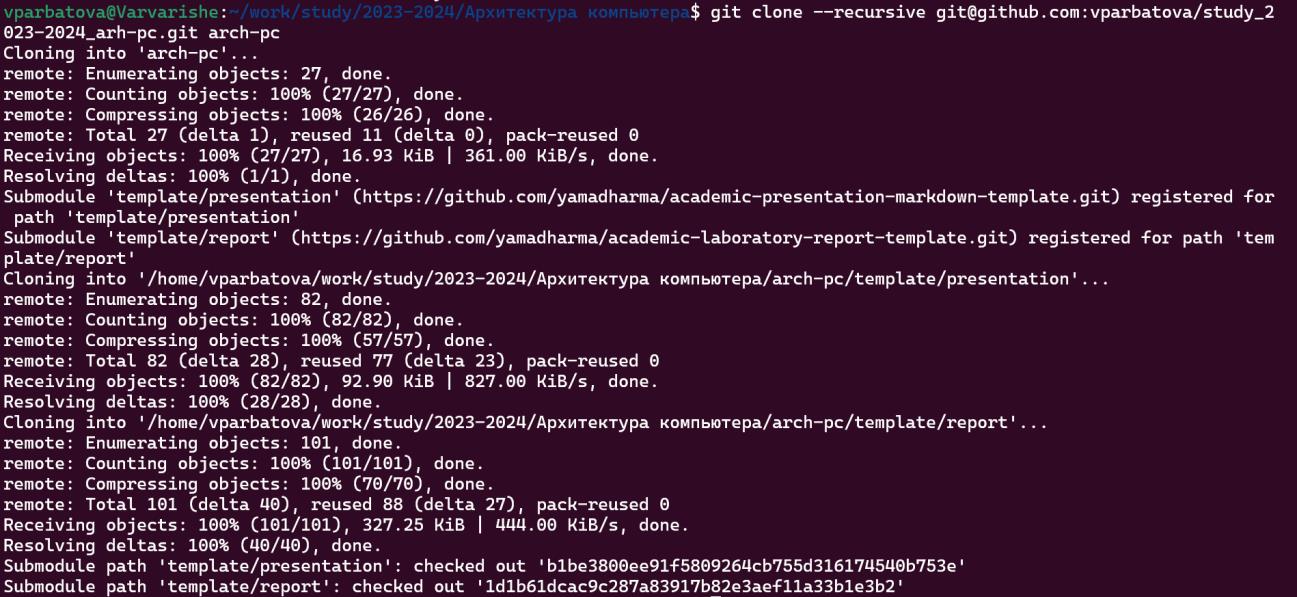


Figure 9: Рис11. Отправка созданных каталогов

С помощью rm удаляю лишние файлы из каталога

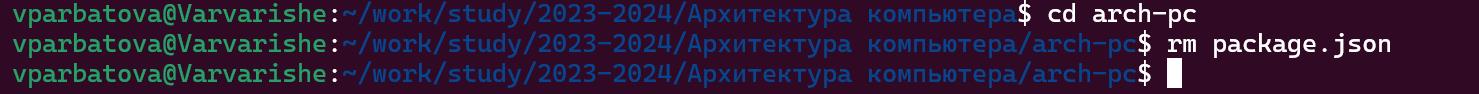


Figure 10: Рис12. Удаление лишних файлов

Создаю необходимые каталоги

Figure 11: Рис13. Создание каталогов

Figure 11: Рис13. Создание каталогов

С помощью git add добавляю все созданные каталоги, затем комментирую и сохраняю изменения на сервере с помощью git commint, и отправляю все на сервер с помощью git push

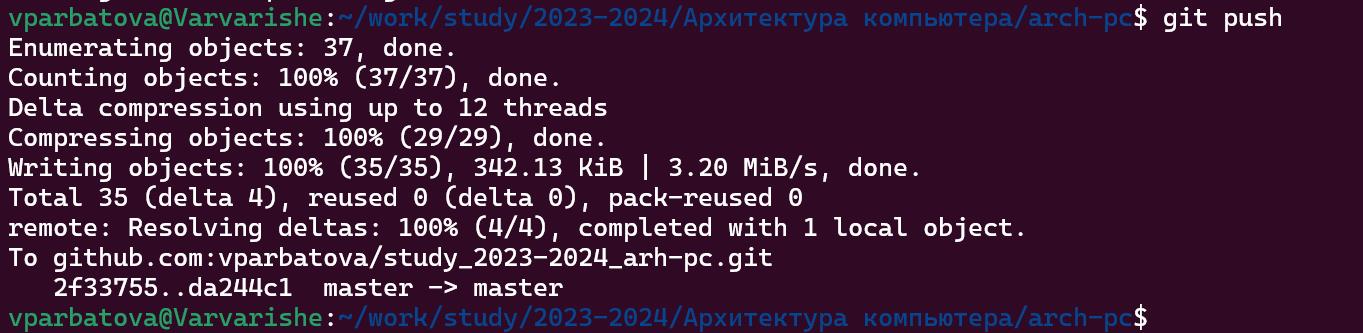


Figure 12: Рис14. Выгрузка изменений на сервер

Проверяю правильность выполнения всех комманд

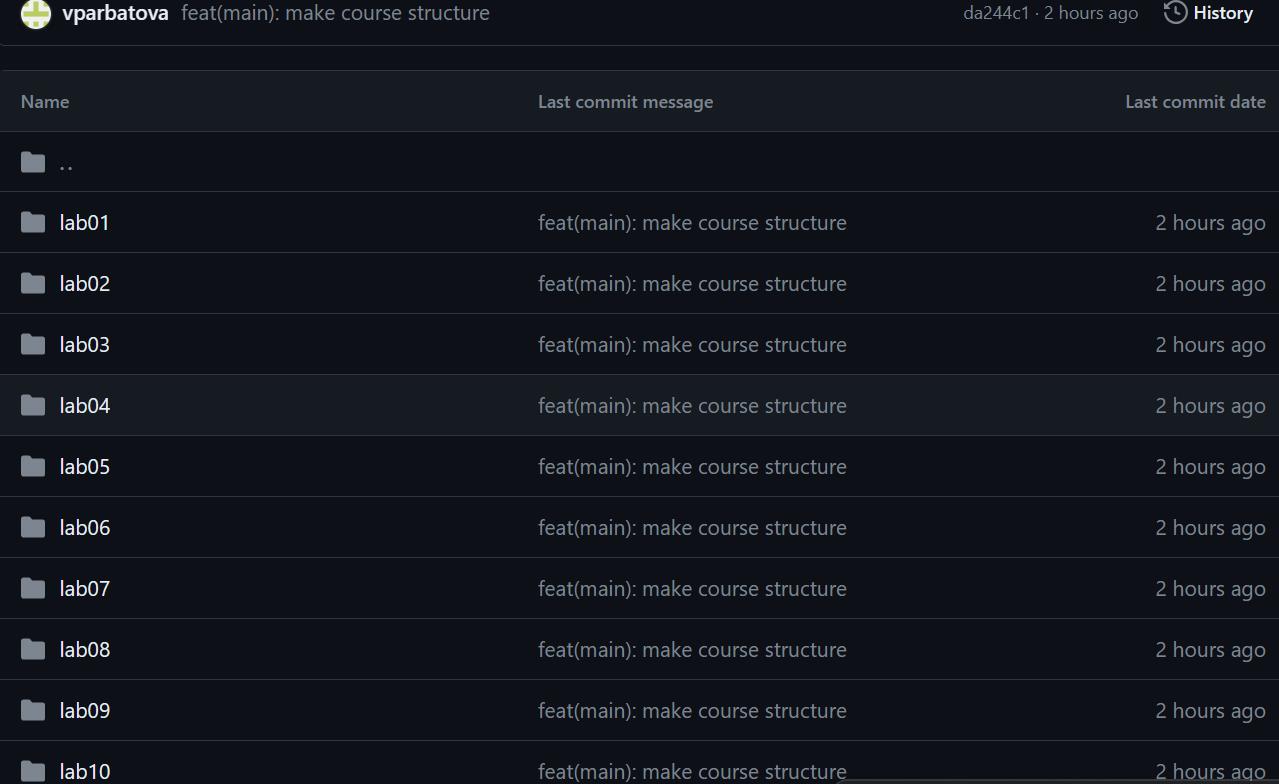


Figure 13: Рис15. Страница репозитория

# Выполнение заданий для самостоятельной работы

Создаю файл в папке

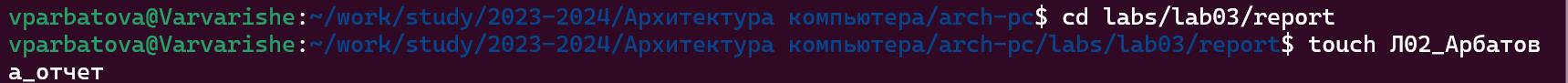


Figure 14: Рис16. Создание файла

Перемещаю файл с ранее выполненной работой 1 и проверяю корректность выполнения

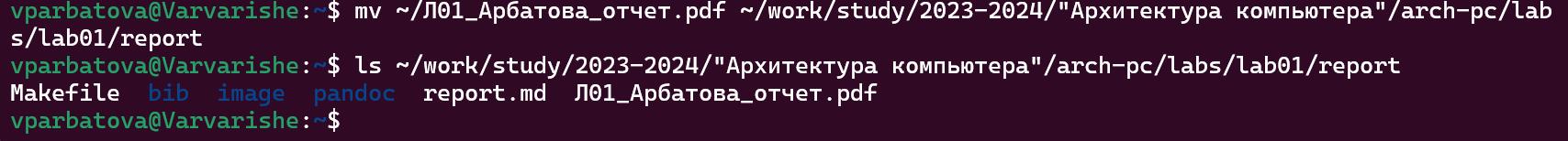


Figure 15: Рис17. Перемещение и проверка файлов

Выгружаю 2 работу на сайт

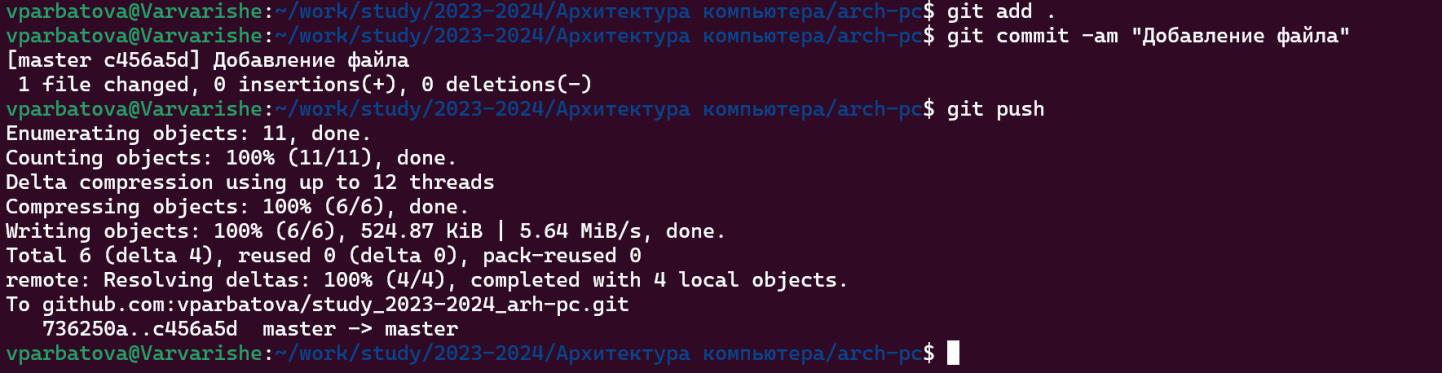


Figure 16: Рис18. Добавление 2 лабораторной

Выгружаю 1 работу на сайт аналогично

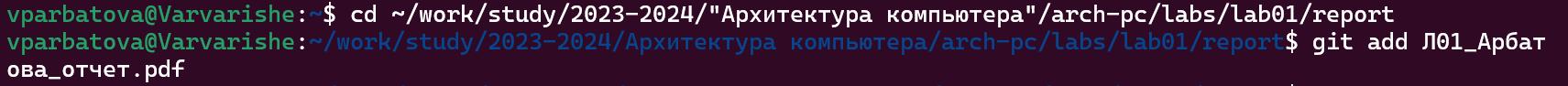


Figure 17: Рис19. Добавление 1 лабораторной

Проверяю выгрузились ли файлы

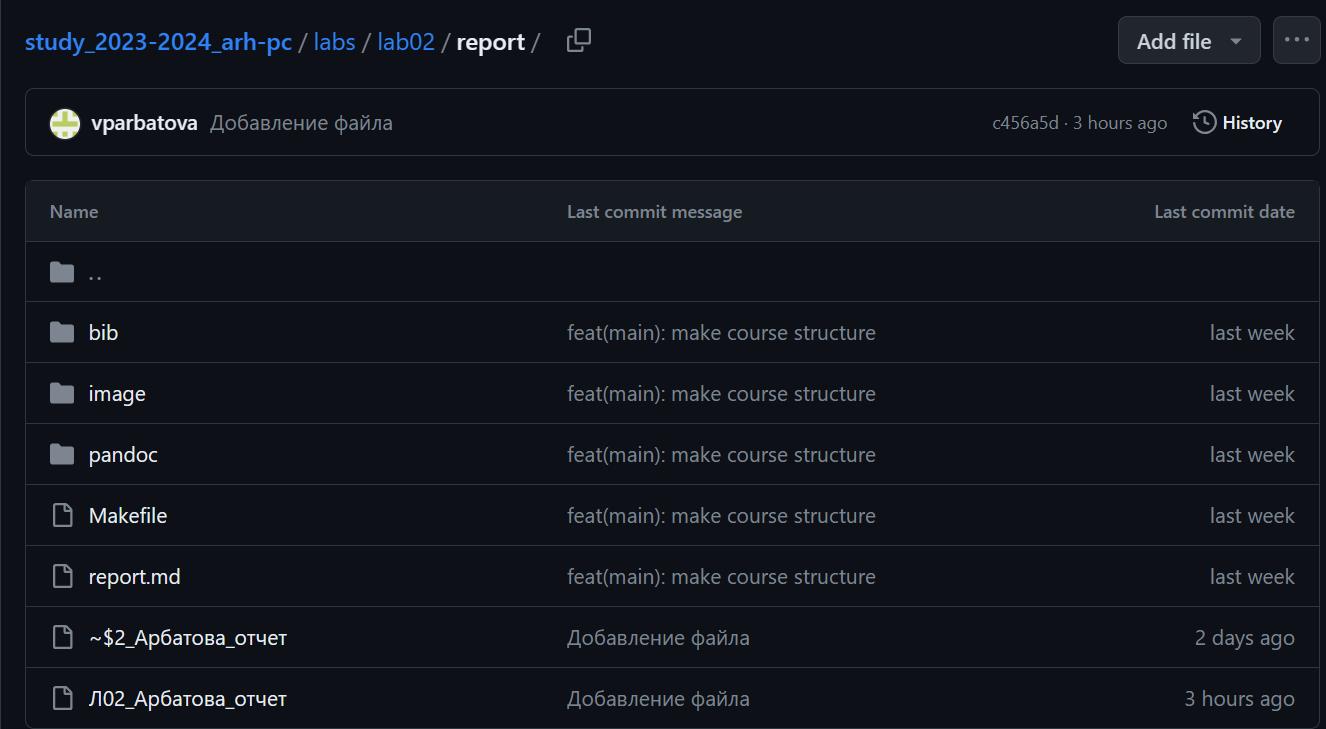


Figure 18: Рис20. Проверка файлов

# Выводы

Я создала репозиторий в github, познакомилась с работой в системе git, выгрузила к себе в аккаунт свои лабораторные работы # Список литературы{.unnumbered}