Отчёт по лабораторной работе №2

Дисциплина: Архитектура компьютера

Ермакова Анастасия Алексеевна

Содержание

Список иллюстраций

Список таблиц

# 1 Цель работы

Цель данной работы – изучение идеологии и применение средств контроля версий, а также приобретение практических навыков по работе с системой git.

# 2 Задание

1. Настройка github
2. Базовая настройка git
3. Создание SSH ключа
4. Сознание рабочего пространства и репозитория курса на основе шаблона
5. Сознание репозитория курса на основе шаблона
6. Настройки каталога курса
7. Выполнение 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 Выполнение лабораторной работы

## 4.1 Настройка github

Для начала создам учетную запись на сайте https://github.com/ и заполню основные данные. (рис. 1)

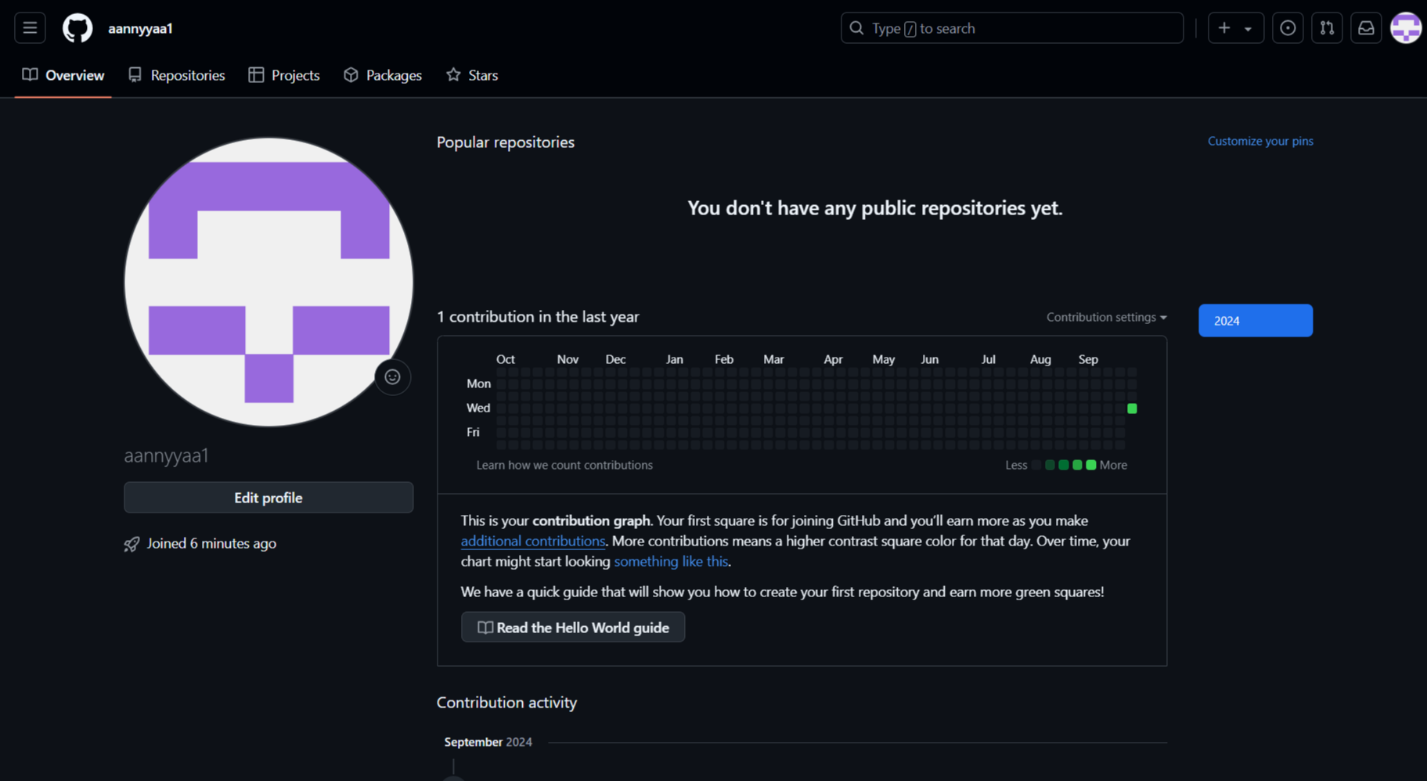


Рис. 1: Учетная запись на сайте GitHub

## 4.2 Базовая настройка git

Делаю предварительную конфигурацию git. Открываю терминал и ввожу следующие команды, указав свое имя и email. (рис. 2)

Рис. 2: Предварительная конфигурация git

Рис. 2: Предварительная конфигурация git

Настраиваю utf-8 в выводе сообщений git для корректного отображения символов. Задаю имя master для начальной ветки. Ввожу параметры autocrlf и safecrlf. (рис. 3)

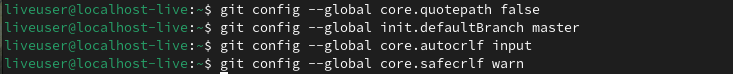


Рис. 3: Настройки git

## 4.3 Создание SSH ключа

Генерирую пару ключей (приватный и открытый) для последующей идентификации на сервере. Для этого ввожу ssh-keygen -C с указанием своего имени и почты. Ключи автоматически сохранятся в каталоге ~/.ssh/. (рис. 4)

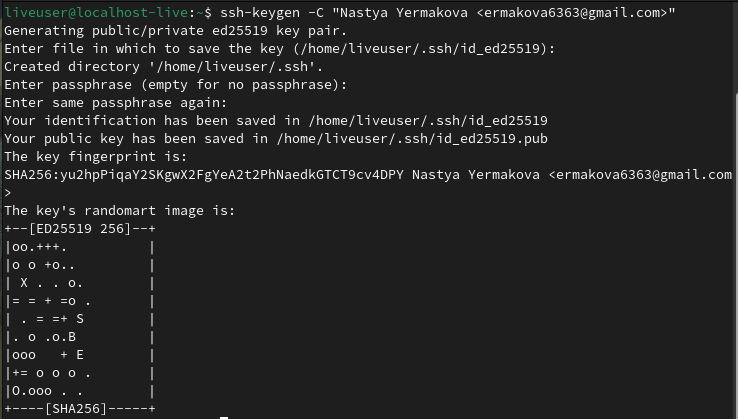


Рис. 4: Создание ключей

Затем захожу на сайт //github.org/ в свою учетную запись, перехожу в меню Settings, в меню SSH and GPG keys нажимаю New SSH key. (рис. 5)

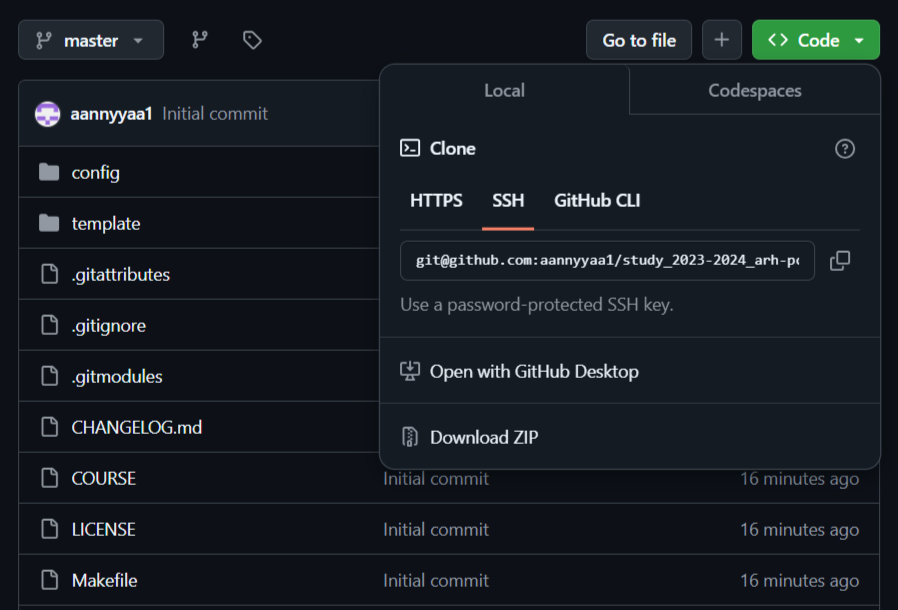


Рис. 5: Окно SSH and GPG keys на сайте

Копирую из локальной консоли ключ в буфер обмена с помощью утилиты xclip. Для этого ее сначала потребуется установить следующим образом: (рис. 6)

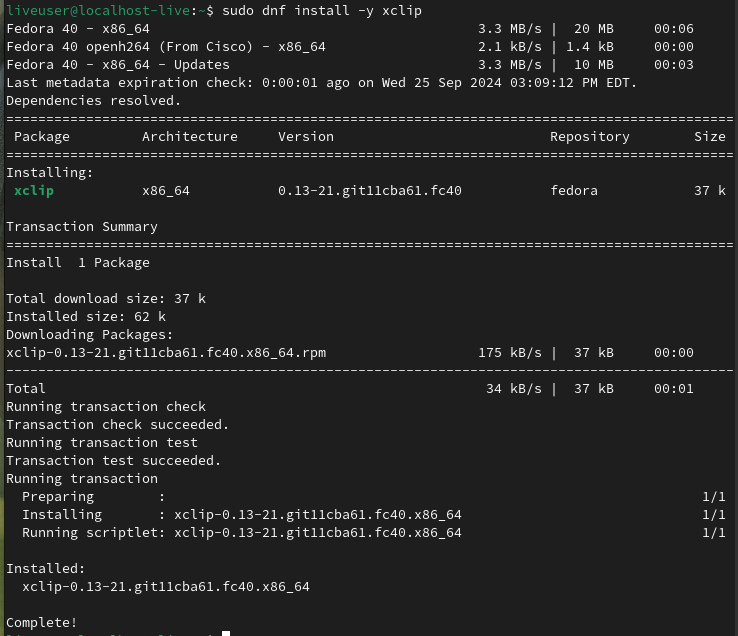


Рис. 6: Установка утилиты xclip

После установки ввожу код, чтобы скопировать ключ, с указанием директории. (рис. 7)

Рис. 7: Копирование ключа

Рис. 7: Копирование ключа

Скопированный ключ вставляю его в поле на сайте и указываю название ключа «Ключ». (рис. 8)

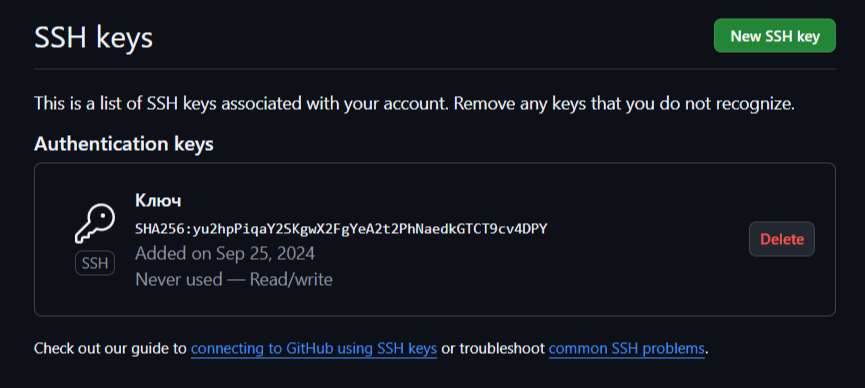


Рис. 8: Ключ на сайте GitHub

## 4.4 Создание рабочего пространства и репозитория курса на основе шаблона

Для того, чтобы при выполнении лабораторных работ придерживаться структуры рабочего пространства, создаю каталог для предмета «Архитектура компьютера». Ввожу следующий код и с помощью команды ls убеждаюсь, что каталог создан. (рис. 9)

Рис. 9: Создание каталога “Архитектура компьютера”

Рис. 9: Создание каталога “Архитектура компьютера”

## 4.5 Создание репозитория курса на основе шаблона

Для создания репозитория буду использовать готовый шаблон курса (https://github.com/yamadharma/course-directory-student-template) на сайте GitHub. Захожу на страницу репозитория и нажимаю Use this template. (рис. 10)

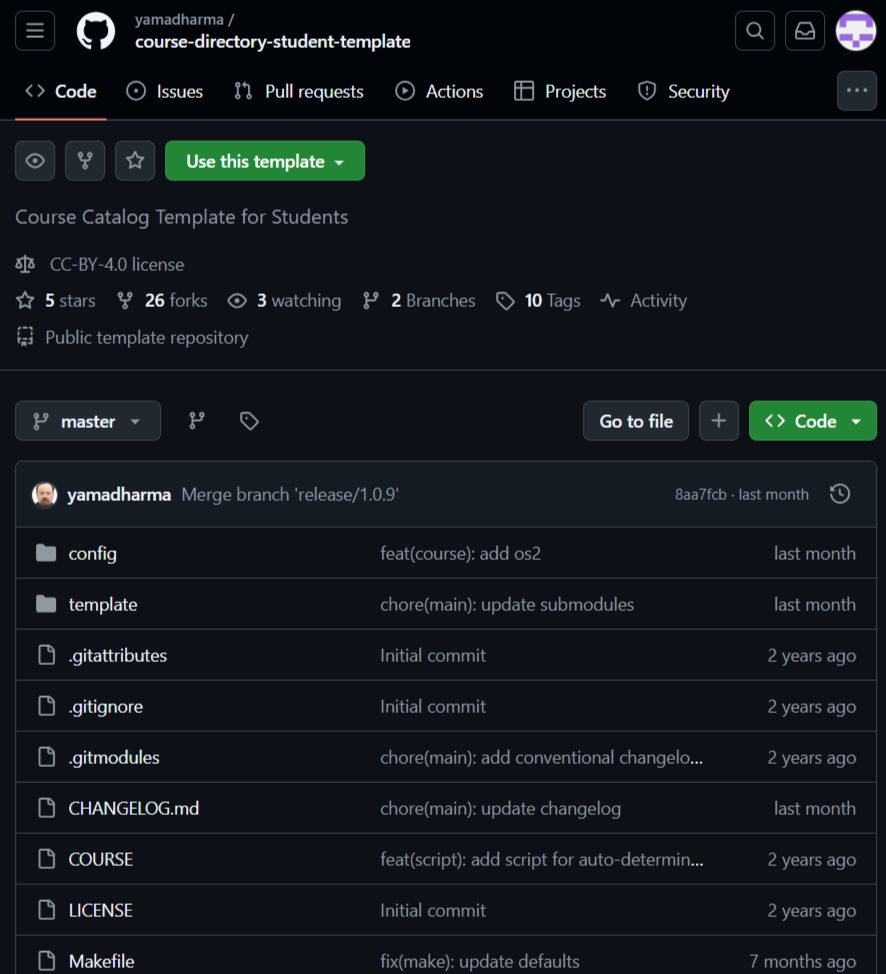


Рис. 10: Страница репозитория курса

В открывшемся окне задаю имя репозитория (study\_2023-2024\_arh-pc) и нажимаю Create repository from template. (рис. 11)

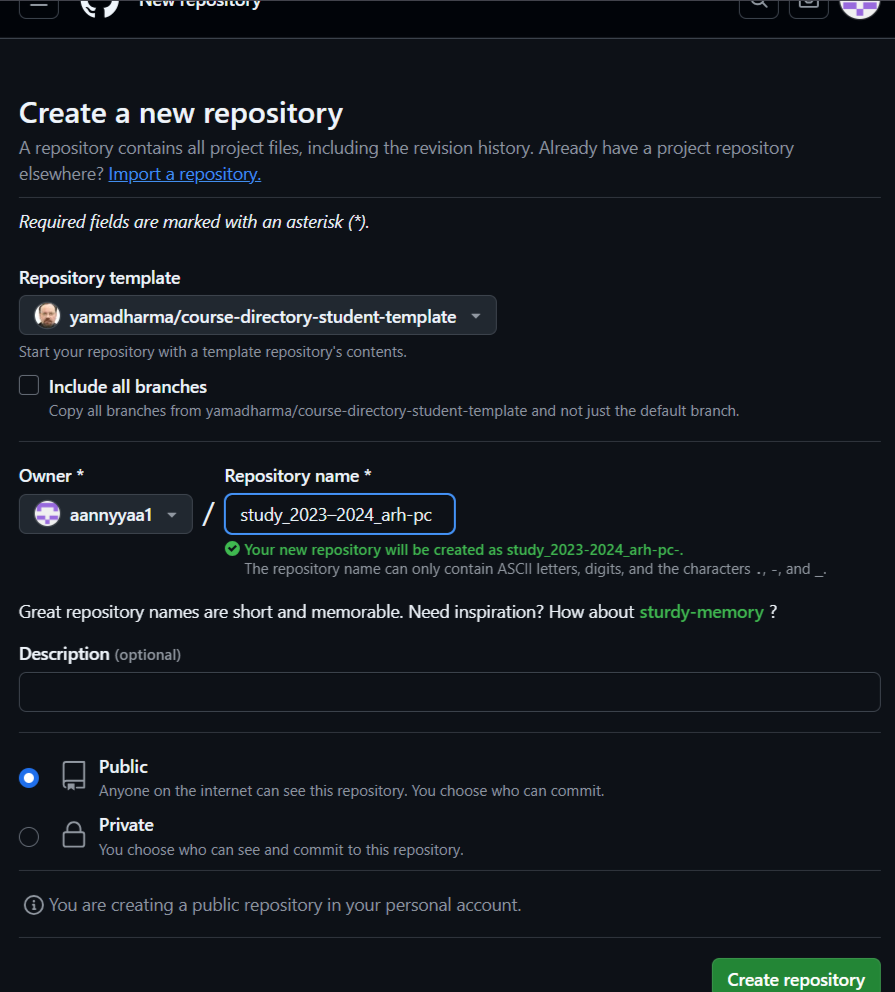


Рис. 11: Создание репозитория на основе шаблона

Репозиторий создан. (рис. 12)

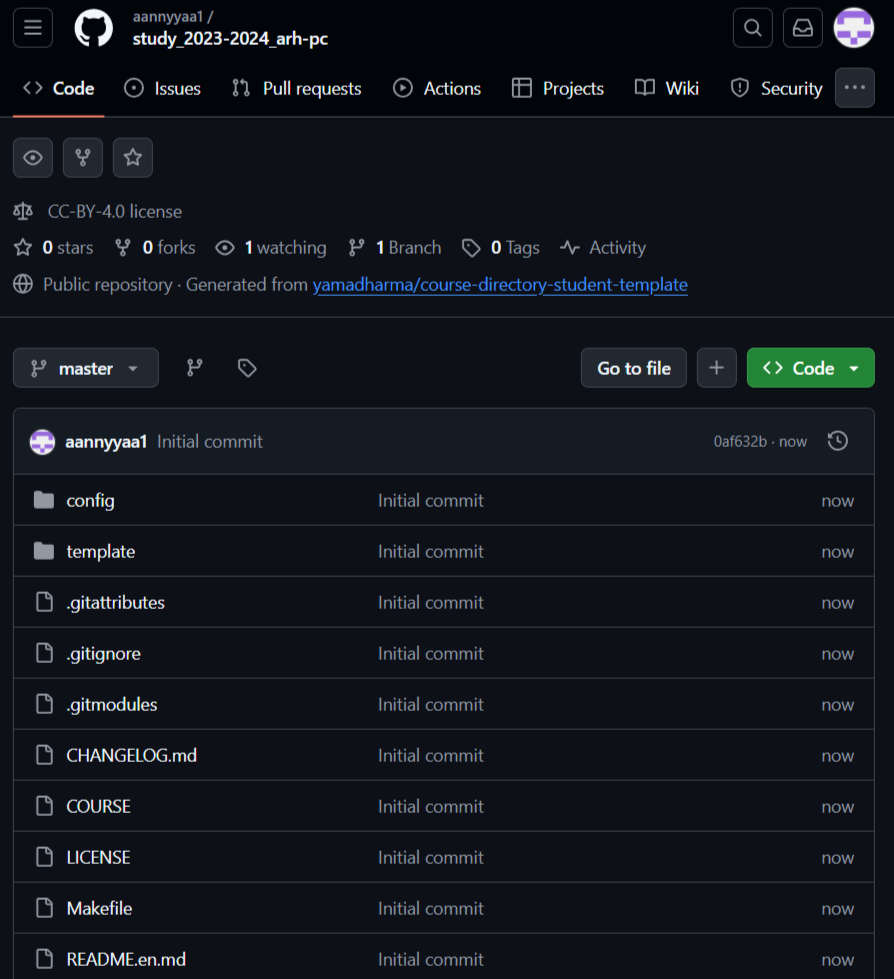


Рис. 12: Созданный репозиторий

Открываю терминал и перехожу в каталог курса с помощью утилиты cd. (рис. 13)

Рис. 13: Перемещение в директорию курса

Рис. 13: Перемещение в директорию курса

Клонирую созданный репозиторий, указав ссылку для клонирования из окна SSH. (рис. 14)

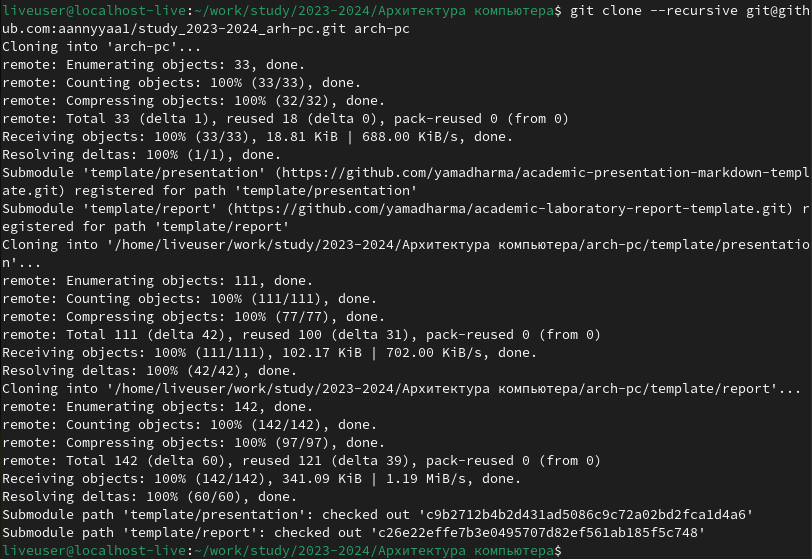


Рис. 14: Клонирование репозитория

## 4.6 Настройка каталога курса

С помощью утилиты cd перехожу в каталог курса. С помощью утилиты rm удаляю ненужные файлы. (рис. 15)

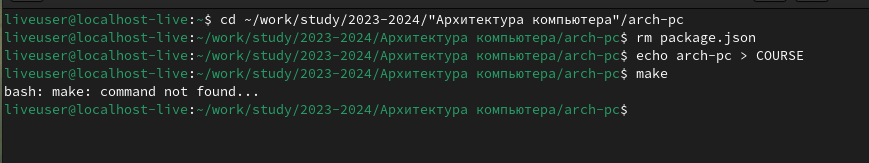


Рис. 15: Удаление ненужных файлов

Создаю необходимые каталоги, ввожу echo arch-pc > Course (рис. 16), затем утилиту make. Данная команда не находится, об этом говорит сообщение «bash: make: command not found…», скачиваю ее. Обновляю базу данных yum с помощью dnf, ввожу команду sudo dnf makecache –refresh. После обновления базы данных скачиваю make с помощью команды sudo dnf -y install make. Команда установлена. (рис. 17)

Рис. 16: Создание каталогов

Рис. 16: Создание каталогов

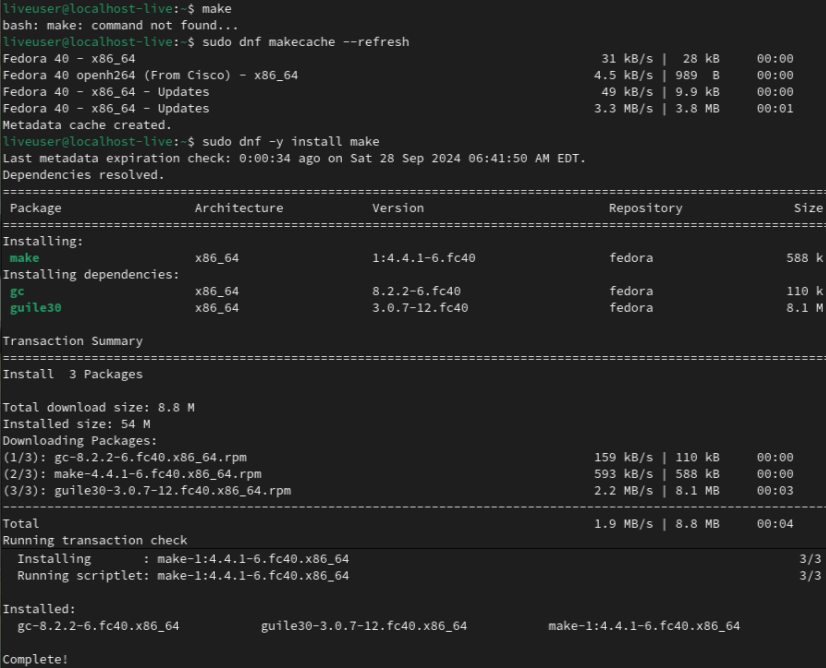


Рис. 17: Установка команды make

После успешной установки использую команду еще раз, но пишу make с таргетом prepare, затем отправляю файлы на сервер. (рис. 18,19)

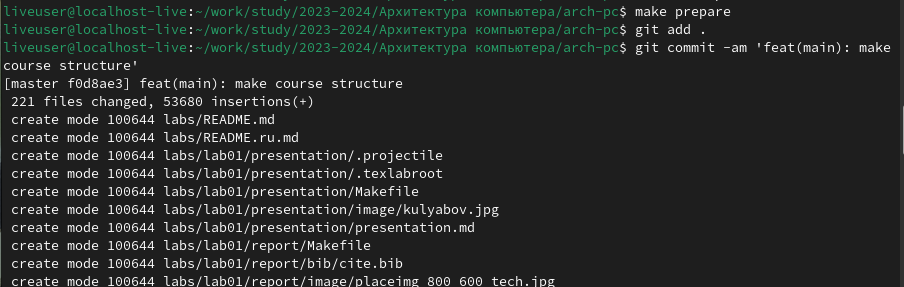


Рис. 18: Создание и отправка на сервер файлов и каталогов

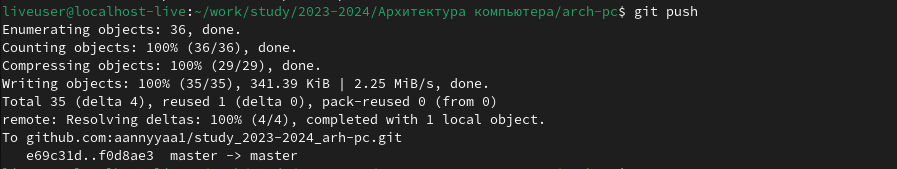


Рис. 19: Создание и отправка на сервер файлов и каталогов

Проверяю правильность создания иерархии рабочего пространства на странице github. (рис. 20)

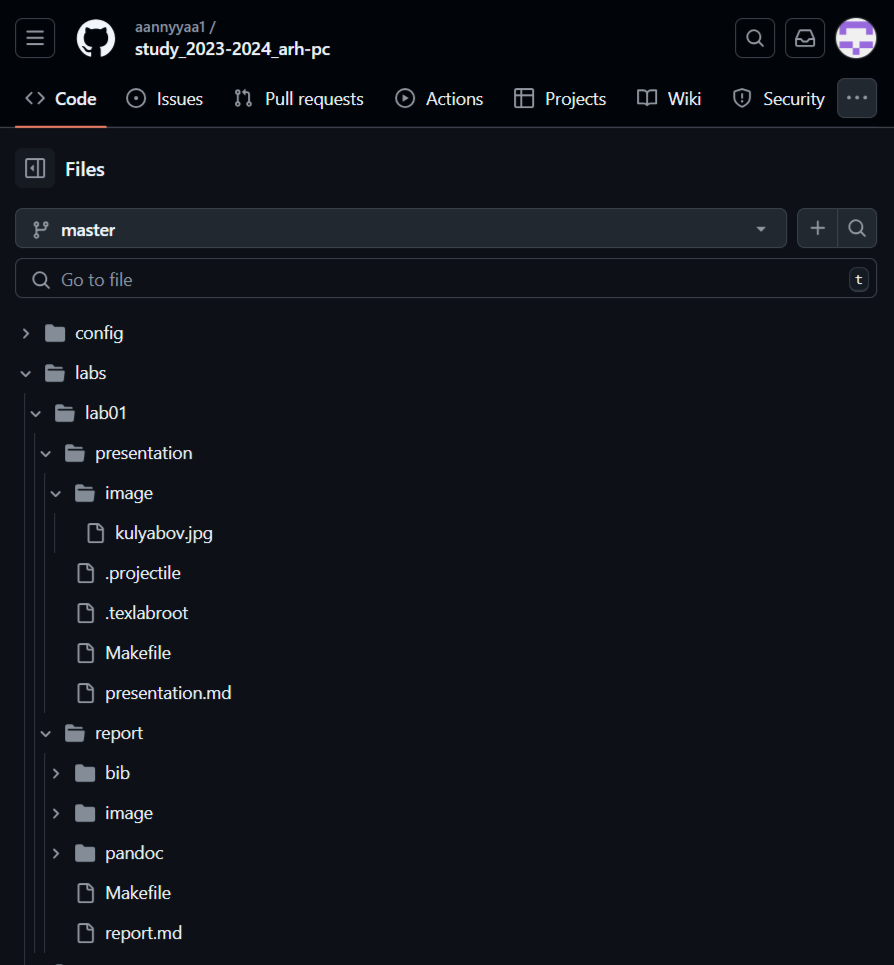


Рис. 20: Иерархия рабочего пространства на странице github

# 5 Выполнение самостоятельной работы

Скопирую отчет по выполнению первой лабораторной работе в каталог labs/lab01/report. Для начала проверю лежит ли сохраненный файл первой лабораторной работы в папке Downloads с помощью команды ls. (рис. 21)

Рис. 21: Расположение файла с отчетом

Рис. 21: Расположение файла с отчетом

Затем с помощью утилиты cp копирую файл в нужный мне каталог. (рис. 22)

Рис. 22: Копирование файла

Рис. 22: Копирование файла

Проверяю правильность выполнения команды с помощью команды ls. (рис. 23)

Рис. 23: Расположение файла в нужном каталоге

Рис. 23: Расположение файла в нужном каталоге

Загружаю файл на сайт github. (рис. 24)

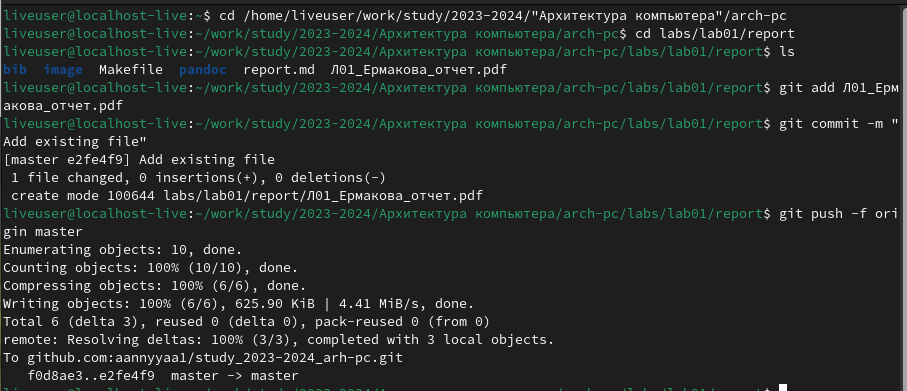


Рис. 24: Загрузка файла на github

Те же самые действия выполняю с файлом данной лабораторной работы. Проверяю содержание файла в каталоге Downloads. Копирую его в нужный каталог. (рис. 25)

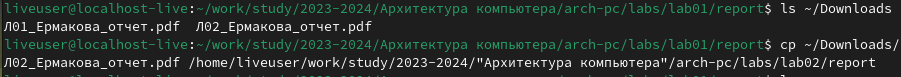


Рис. 25: Расположение файла отчета №2. Копирование файла

Проверяю, находится ли теперь файл в каталоге labs/lab02/report с помощью команды ls. (рис. 26)

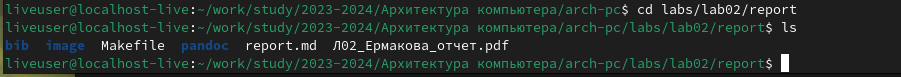


Рис. 26: Расположение файла в нужном каталоге

И наконец загружаю файл на сайт github. (рис. 27)

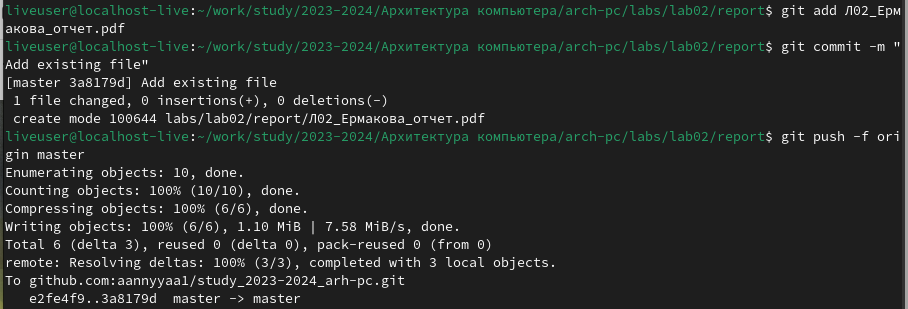


Рис. 27: Загрузка файла на сайт

Непосредственно на сайте проверяю, все ли программы были выполнены верно. Да, файлы с отчетами загрузились в нужные каталоги. (рис. 28)

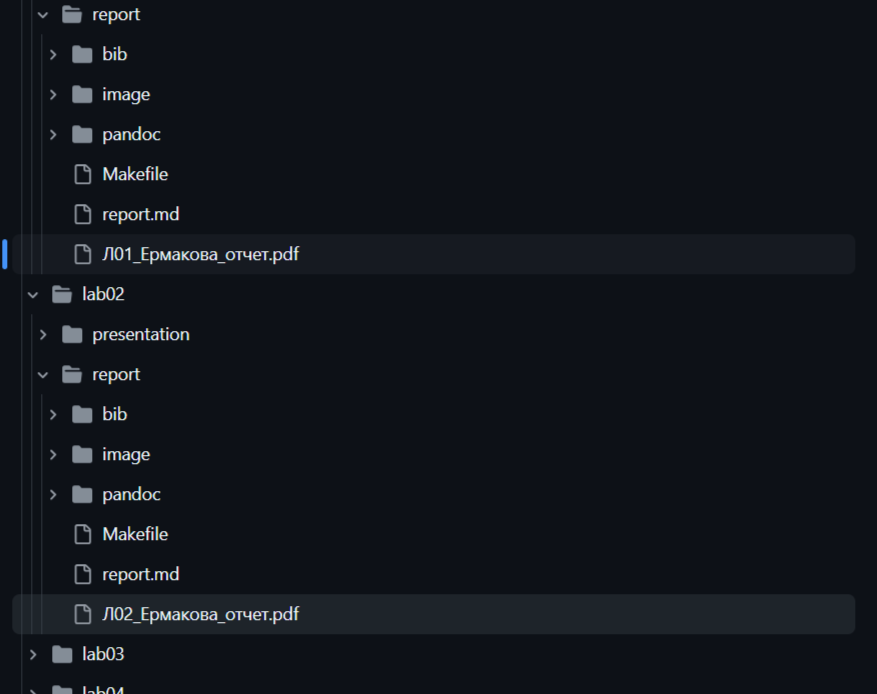


Рис. 28: Репозиторий на сайте

# 6 Выводы

В ходе выполнения данной лабораторной работы я приобрела практические навыки по работе с системой git, изучила идеологию и применение средств контроля версий.

# 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)