Отчет по лабораторной работе №2

Дисциплина: Архитектура компьютера

Малюга Валерия Васильевна

Содержание

# 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 и репозитория курса на основе шаблона.
5. Создание репозитория курса на основе шаблона.
6. Настройка каталога курса.
7. Выполнение заданий 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. 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. 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. 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. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным. 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.

Основные команды git{#tbl:std-dir}

| Команда | Описание |
| --- | --- |
| git init | создание основного дерева репозитория |
| git pull | получение обновлений (изменений) текущего дерева из центрального репозитория |
| git push | отправка всех произведённых изменений локального дерева в центральный репозиторий |
| git status | просмотр списка изменённых файлов в текущей директории |
| git diff | просмотр текущих изменения |
| git add | добавить все изменённые и/или созданные файлы и/или каталоги |
| git add имена\_файлов | добавить конкретные изменённые и/или созданные файлы и/или каталоги |
| git rm имена\_файлов | удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории) |
| git commit -am 'Описание коммита' | сохранить все добавленные изменения и все изменённые файлы |
| git checkout -b имя\_ветки | создание новой ветки, базирующейся на текущей |
| git checkout имя\_ветки | переключение на некоторую ветку (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой) |
| git push origin имя\_ветки | отправка изменений конкретной ветки в центральный репозиторий |
| git merge --no-ff имя\_ветки | слияние ветки с текущим деревом |
| git branch -d имя\_ветки | удаление локальной уже слитой с основным деревом ветки |
| git branch -D имя\_ветки | принудительное удаление локальной ветки |
| git push origin :имя\_ветки | удаление ветки с центрального репозитория |

# 3 Выполнение лабораторной работы

## 3.1 Настройка GitHub

Создала учетную запись на сайте https://github.com/(рис. 1).

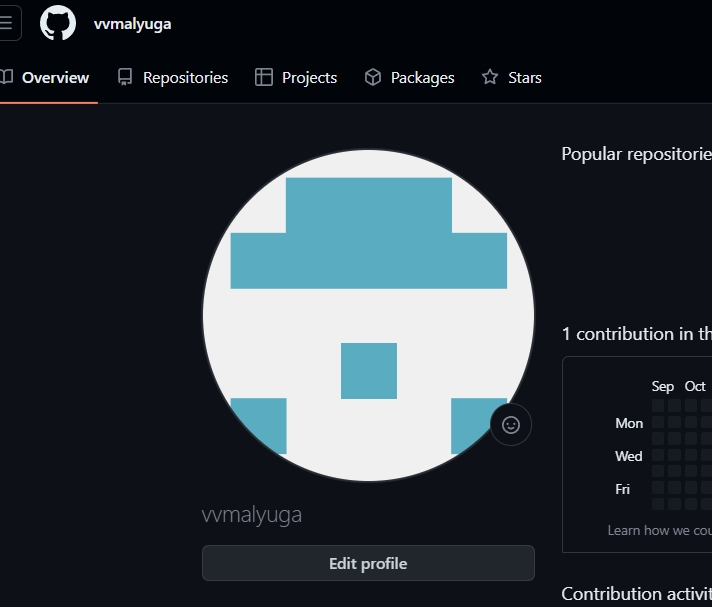


Рис. 1: Профиль в github

## 3.2 Базовая настройка Git

Открыла терминал и ввела следующие команды, указав мои имя и email. Настроила utf-8 в выводе сообщений git. Задала имя начальной ветки (master) и параметры autocrlf, safecrlf (рис. 2).

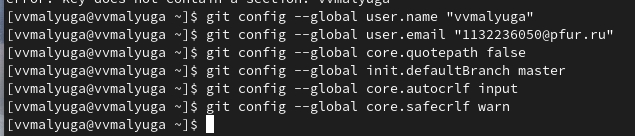


Рис. 2: Настройка git

## 3.3 Создание SSH-ключа

Сгенерировала пару ключей (приватный и открытый) (рис. 3). Просмотрела ключ с помощью команды cat, скопировала его, вставила в поле для ключа на сайте (рис. 4). Авторизовала ключ (рис. 5).

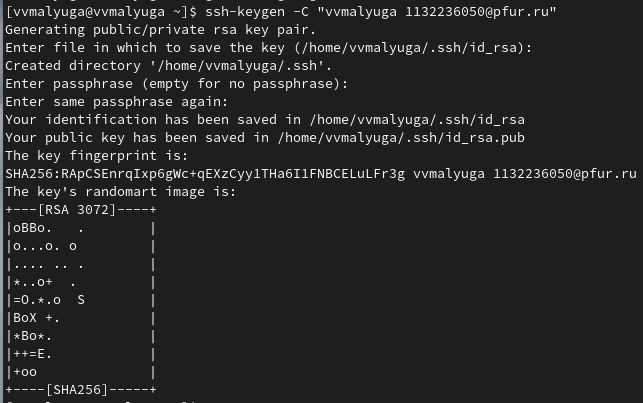


Рис. 3: Создание SHH ключа

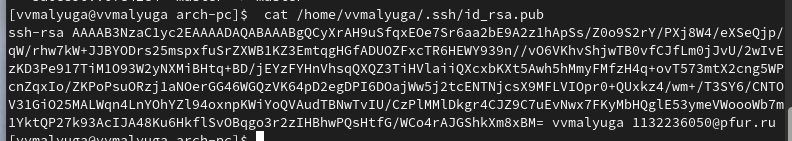


Рис. 4: Открытый ключ SHH

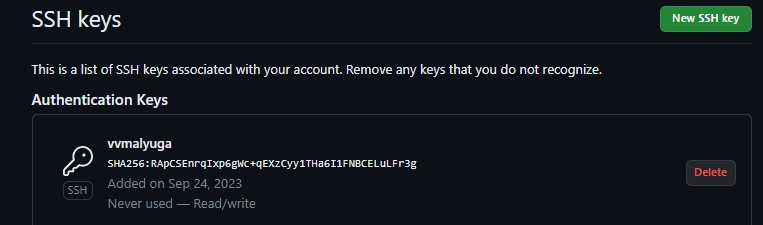


Рис. 5: Лист авторизованных ключей

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

Открыла терминал и создала каталог для предмета «Архитектура компьютера». Проверила с помощью команды ls (рис. 6).

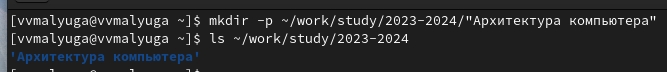


Рис. 6: Создание каталога «Архитектура компьютера» и проверка

## 3.5 Создание репозитория курса на основе шаблона

Перехожу на страницу репозитория с шаблоном курса https://github.com/yamadharma/course-directory-student-template. Далее выбрала Use this template (рис. 7). В открывшемся окне задала имя репозитория study\_2023–2024\_arhpc и создала репозиторий (рис. 8).

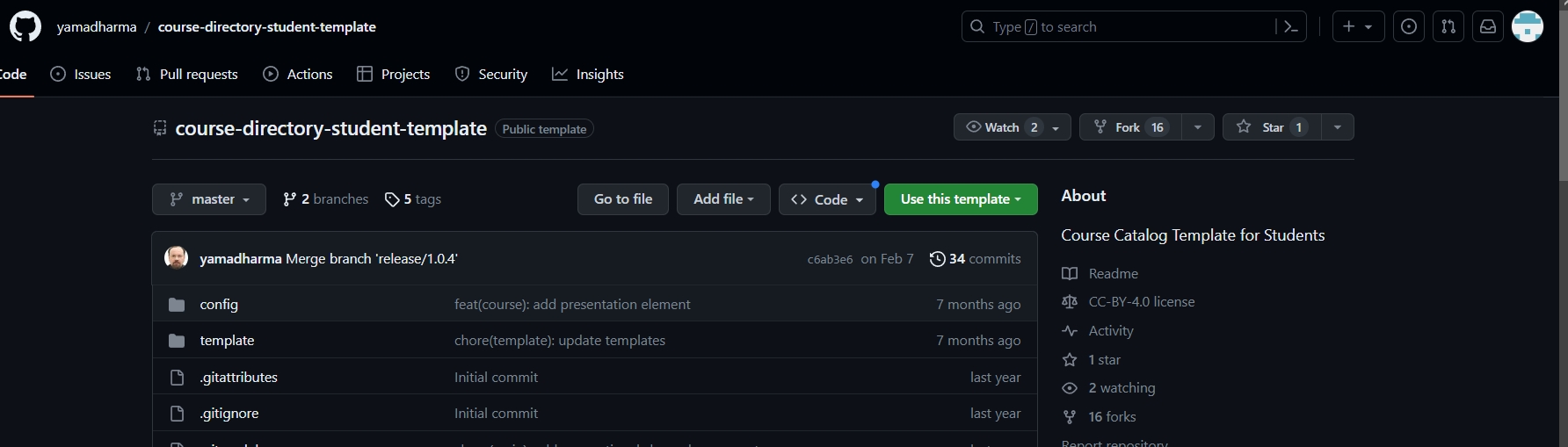


Рис. 7: Страница репозитория с шаблоном курса

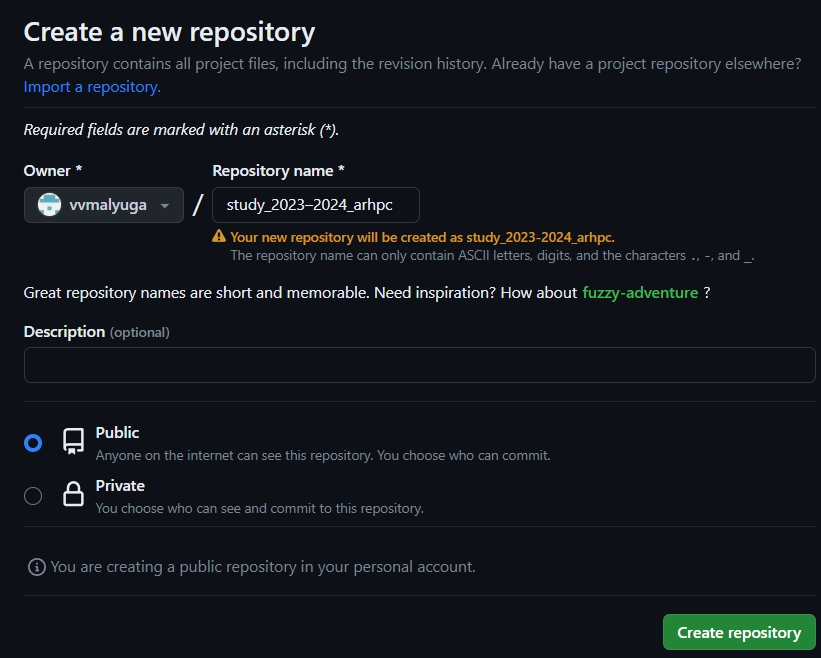


Рис. 8: Страница создания репозитория study\_2023–2024\_arhpc

Открыла терминал и перешла в каталог курса. Клонировала созданный репозиторий, скопировав ссылку для клонирования на странице созданного репозитория Code -> SSH (рис. 9).

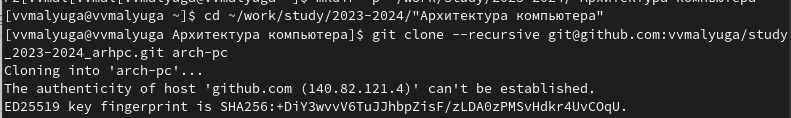


Рис. 9: Клонирование созданного репозитория

## 3.6 Настройка каталога курса

Перешла в каталог курса, удалила лишние файлы с помощью команды rm (рис. 10).

Рис. 10: Удаление лишних файлов

Рис. 10: Удаление лишних файлов

Создала необходимые каталоги с помощью git add, комментирую и сохраняю изменения на сервере как добавление курса с помощью git commit (рис. 11). Отправила файлы на сервер с помощью push (рис. 12). Проверила правильность создания иерархии рабочего пространства в локальном репозитории и на странице github (рис. 13).

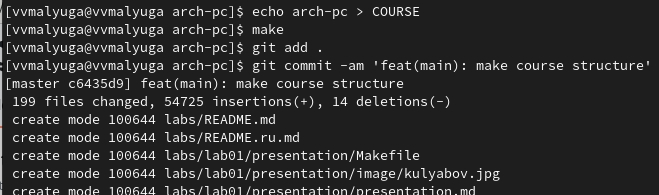


Рис. 11: Создание необходимых каталогов

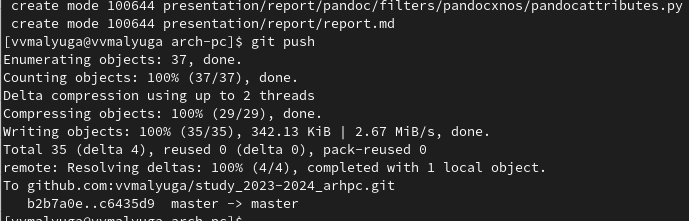


Рис. 12: Отправка файлов на сервер

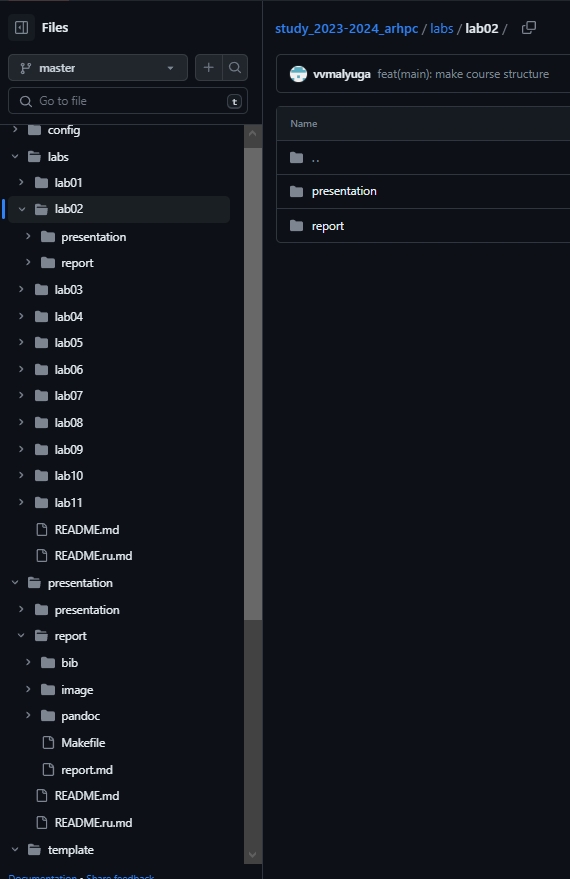


Рис. 13: Страница репозитория

## 3.7 Выполнение заданий для самостоятельной работы

1. Перехожу в директорию labs/lab02/report с помощью утилиты cd. Создаю в каталоге файл для отчета по второй лабораторной работе с помощью утилиты touch (рис. 14).

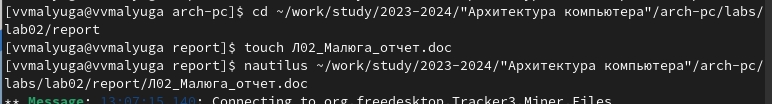


Рис. 14: Создание текстового документа для оформления отчета

Оформляю отчет по лабораторной работе №2 в текстовом процессоре LibreOffice Writer.

1. Первая лабораторная работа у меня находится в подкаталоге домашней директори Downloads. Перехожу в каталог […]/arch-pc/labs/lab01/report и копирую сюда файл Л01\_Малюга\_отчет.pdf с помощью команды cp и проверяю выполнение с помощью ls (рис. 15).

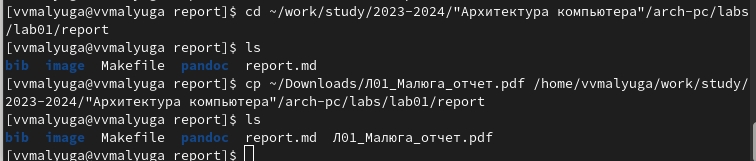


Рис. 15: Копирование отчета по лабораторной работе №1 в каталог

1. Перешла в каталог arch-pc, чтобы отправить все добавленные файлы в репозиторий. Использую команды git add (добавляю все измененные файлы), и затем git commit -m «Add existing files» (сохраняю все измененные файлы), и в конце git push, чтобы отправить все изменения на сервер (рис. 16).

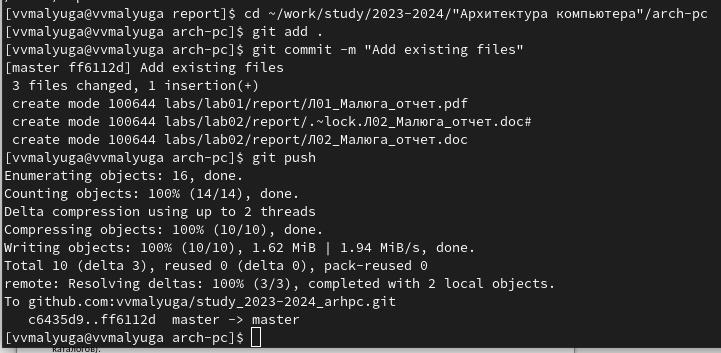


Рис. 16: Добавление и отправка файлов в центральный репозиторий

Проверила выполнение работы на сайте github. Действительно, все добавилось (рис. 17).

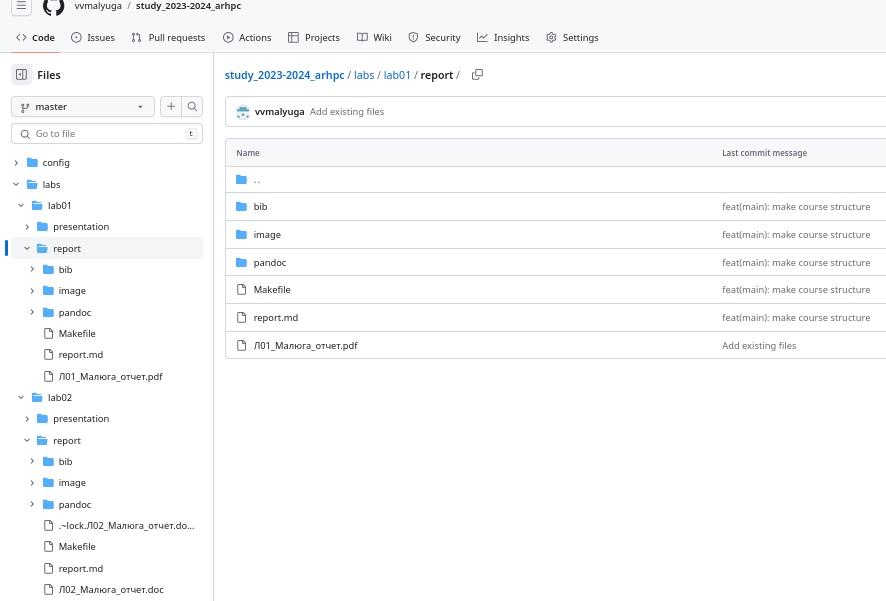


Рис. 17: Страница с добавленными файлами в подкаталогах lab01 и lab02

# 4 Выводы

Изучила идеологию и средства контроля версий, также приобрела практические навыки по работе с системой git.