РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

дисциплина: Архитектура компьютера

Студент: Томилова Валентина

Группа: НКАбд-06-25

МОСКВА

2025 г.

Оглавление

Цель работы 3
Теоретическое ведение 4
2.1 Системы контроля версий. Общие понятия 4
2.2 Система контроля версий Git 4
2.2 Основные команды git 5
Выполнения лабораторной работы 7
3.1 Настройка github 7
3.2 Базовая настройка git 8
3.3 Создание SSH-ключа 8
3.4 Создание рабочего пространства и репозитория курса 10
3.5 Настройка каталога курса 11
Выполнение самостоятельной работы 12
Выводы 13

Цель работы

Целью работы является изучение идеологии и применения средств контроля версий, приобретение практических навыков по работе с системой контроля версий git.

Теоретическое ведение

2.1 Системы контроля версий. Общие понятия

Что такое «**система контроля версий**» и почему это важно? **Система контроля версий** — это система, записывающая изменения в файл или набор файлов в течение времени и позволяющая вернуться позже к определённой версии. Для контроля версий файлов в этой книге в качестве примера будет использоваться исходный код программного обеспечения, хотя на самом деле вы можете использовать контроль версий практически для любых типов файлов.

Если вы графический или web-дизайнер и хотите сохранить каждую версию изображения или макета (скорее всего, захотите), система контроля версий (далее VCS) — как раз то, что нужно. Она позволяет вернуть файлы к состоянию, в котором они были до изменений, вернуть проект к исходному состоянию, увидеть изменения, увидеть, кто последний менял что-то и вызвал проблему, кто поставил задачу и когда и многое другое. Использование VCS также значит в целом, что, если вы сломали что-то или потеряли файлы, вы спокойно можете всё исправить. В дополнение ко всему вы получите всё это без каких-либо дополнительных усилий. [1]

2.2 Система контроля версий Git

Git — это распределённая система управления версиями , которая позволяет командам разработчиков программного обеспечения иметь несколько локальных копий кодовой базы проекта, независимых друг от друга. Эти копии, или ветви, можно быстро создавать, объединять и удалять, что позволяет командам экспериментировать с минимальными вычислительными затратами, прежде чем объединить их в основную ветку (иногда называемую главной веткой). Git известен своей скоростью, совместимостью с рабочими процессами и открытым исходным кодом.[2]

Рисунок 1. Система контроля версий

2.2 Основные команды git

В данной таблице будут предоставлены все основные команды **Git**.[3]

Основные команды	Описание команд
git init	Инициализация репозитория
git config	Настройки параметров конфигурации Git.
git status	Состояние рабочего каталога и индекса.
git add	Добавление изменений в индекс.
git reset	Отмена изменений в репозитории.
git commit	Сохранение изменений в локальном репозитории Git.
git log	Просмотр истории коммитов
git push	Отправка коммитов в удаленный репозиторий
git branch	Управление ветками
git switch	Переключение между ветками

git clone	Создание копии удаленного репозитория
git stash	Временное хранилище
git config alias	Создание псевдонимов
git checkout	Работа с ветками, восстановление файлов и переключение на конкретные коммиты
git merge	Слияние изменений веток
git fetch	Загрузка обновлений из удаленного репозитория
git pull	Извлечение и слияние изменения
git rebase	Ашалеть, что это за ЗВЕРЬ?
git diff	Просмотр различий между файлами
git difftool	Просмотр различий и редактирование файлов
git remote	Работа с удалёнными репозиториями
git tag	Теги
git restore	Восстановления файлов из индекса или коммитов
git cherry-pick	Применение коммитов из одной ветки на другую
git revert	Откат изменений

Таблица 1. Основные команды

Выполнения лабораторной работы

3.1 Настройка github

Первым шагом было создание учётной записи на сайте GitHub, который будет использоваться как удаленный сервер для хранения репозиториев.

Инструкция по выполнению:

1. Перейдите на сайт https://github.com/.
2. Нажмите на кнопку **"Sign up".**
3. Следуйте инструкциям на экране: введите вашу электронную почту, создайте пароль и выберите имя пользователя.
4. Подтвердите свою учетную запись через письмо, которое придет на указанную почту.

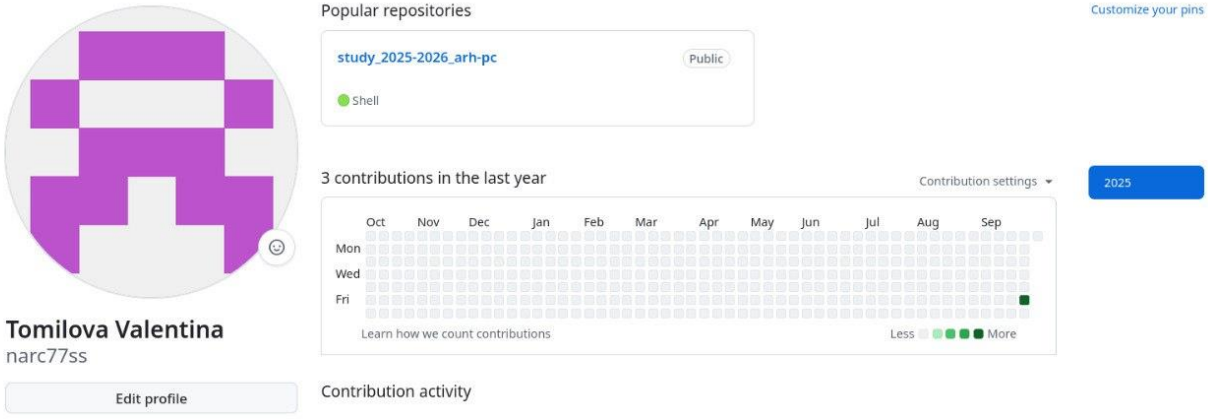


Рисунок 2. Мой профиль на **GitHub**

Я успешно зарегистрировалась на GitHub. Процесс оказался простым и интуитивно понятным.

3.2 Базовая настройка git

После установки **Git** на локальной машине необходимо было выполнить базовую конфигурацию: указать имя пользователя и адрес электронной почты, которые будут отображаться в истории коммитов, а также настроить другие параметры для корректной работы.

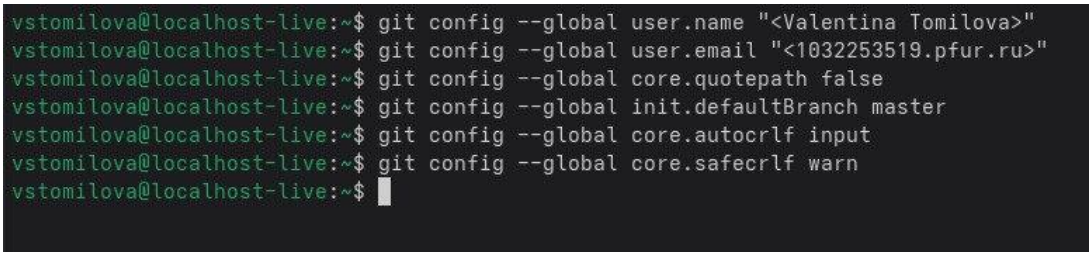


Рисунок 2. Настройка **Git**

Я выполнила базовую настройку **Git**. Эти команды нужны, чтобы каждый мой коммит (сохранение изменений) был подписан моим именем и почтой. Это очень важно для отслеживания истории изменений, особенно при работе в команде.

3.3 Создание SSH-ключа

Для безопасного подключения к GitHub без необходимости каждый раз вводить пароль, я сгенерировала пару SSH-ключей (приватный и публичный) и добавила публичный ключ в свой профиль на GitHub.

```
vstomilova@localhost-live:~$ ssh-keygen -C "Valentina Tomilova <1032253519@pfur.ru>"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/vstomilova/.ssh/id_ed25519):
Created directory '/home/vstomilova/.ssh'.
Enter passphrase for "/home/vstomilova/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/vstomilova/.ssh/id_ed25519
Your public key has been saved in /home/vstomilova/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:2t4ngoce7FHmQvV+OE5lgbWQ/5qPMJ23yFfhtrT1dRs Valentina Tomilova <1032253519@pfur.ru>
The key's randomart image is:
+--[ED25519 256]--+
|          ...    |
|          .+ .   |
|        . ..o    |
|       . .  .. . |
|      . S . o.. .|
|     o * . = ..E=|
|      *oo B +o+.X|
|     .o=o+.*++.=.|
|     .o...oo+oo  |
+----[SHA256]-----+
vstomilova@localhost-live:~$
```

Рисунок 3. Генерация ключи

Все, мы создали ключ, теперь его можно скопировать данной командой:

```
vstomilova@localhost-live:~$ cat ~/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIIY67qiO4MhOGn+EhaYIe2uZmd1LWPYE20fMnH28gDeo Valentina To
milova <1032253519@pfur.ru>
```

Рисунок 4. Копирование ключа

Теперь давайте добавим ключ на **GitHub**:

1. Зайдем в свой профиль на **GitHub**.
2. Перейдем в **Settings -> SSH and GPG keys**.
3. Нажмем **New SSH key**.
4. В поле **Title** введем название ключа (например, "**Home**"), а в поле **Key** вставим скопированный ключ.
5. Нажмите **Add SSH key**.

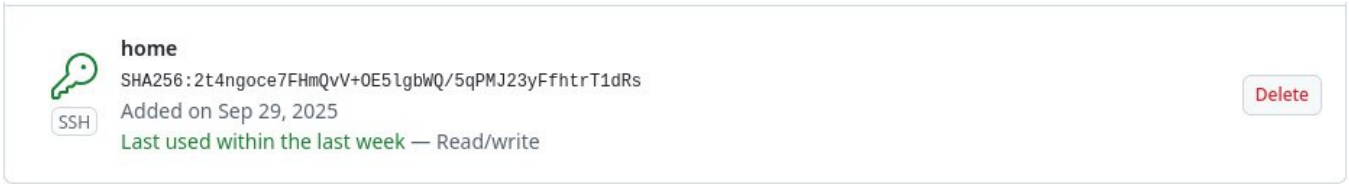


Рисунок 5. SSH key в GitHub

Я сгенерировала SSH-ключ. Это позволяет мне безопасно подключаться к моему репозиторию на GitHub.

3.4 Создание рабочего пространства и репозитория курса

На этом этапе я создала репозиторий на **GitHub** на основе предоставленного шаблона, затем создала локальную структуру каталогов для учебных проектов и клонировала удаленный репозиторий на свой компьютер.

1. Я перешла на страницу репозитория с шаблоном курса: https://github.com/yamadharma/course-directory-student-template.
2. Нажала на кнопку **Use this template**.
3. В открывшемся окне задала имя репозитория **study_2025–2026_arch-pc** и создала репозиторий, нажав кнопку **Create repository from template**.

Теперь создадим каталог и перейдем в него:

```
vstomilova@localhost-live:~$ mkdir -p ~/work/study/2025-2026/"Архитектура компьютера"
vstomilova@localhost-live:~$ cd ~/work/study/2025-2026/"Архитектура компьютера"
```

Рисунок 6. Новый каталог

Клонируем созданный удаленный репозиторий на свой компьютер, скопировав SSH-ссылку со страницы репозитория.

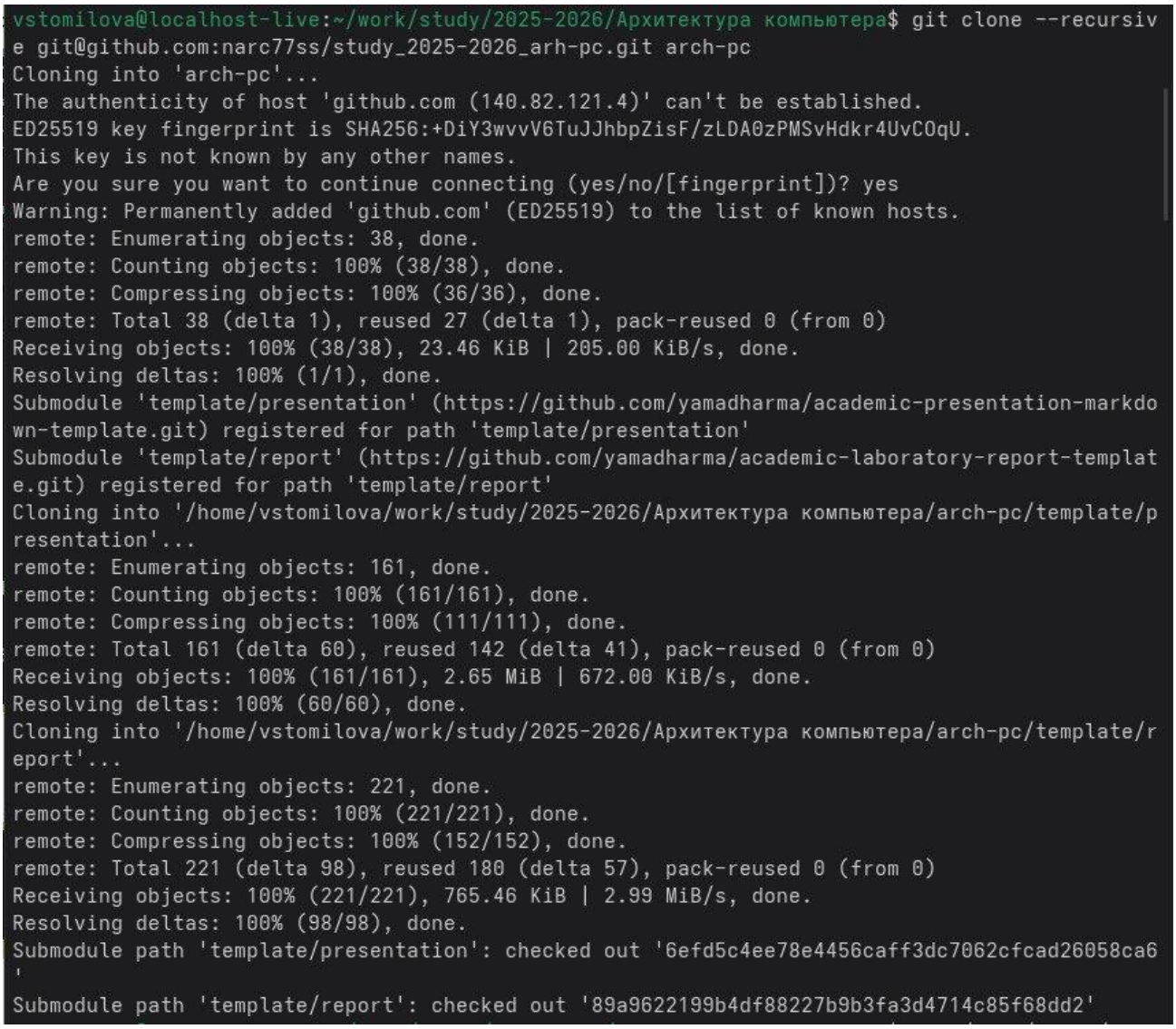


Рисунок 7. Клонирование

Сначала я создала репозиторий на **GitHub**, используя веб-интерфейс и готовый шаблон. Затем я создала локальную структуру папок на своем компьютере и клонировала туда удаленный репозиторий.

3.5 Настройка каталога курса

Завершающим этапом основной части работы была настройка структуры каталога курса с помощью make и отправка начальных файлов на сервер **GitHub**.

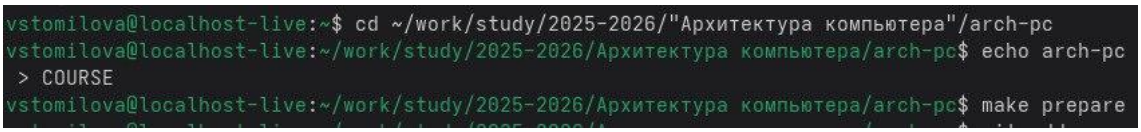


Рисунок 8. Подготовка файлов

Сейчас я перешла в каталог с проектом и выполнила подготовку структур.

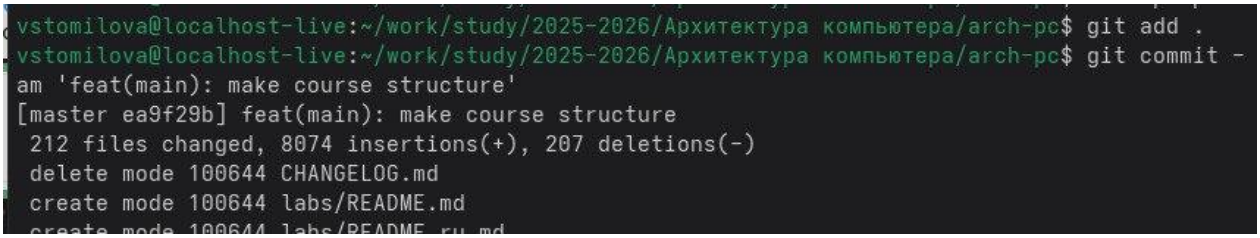


Рисунок 9. Выполнение команд add и commit

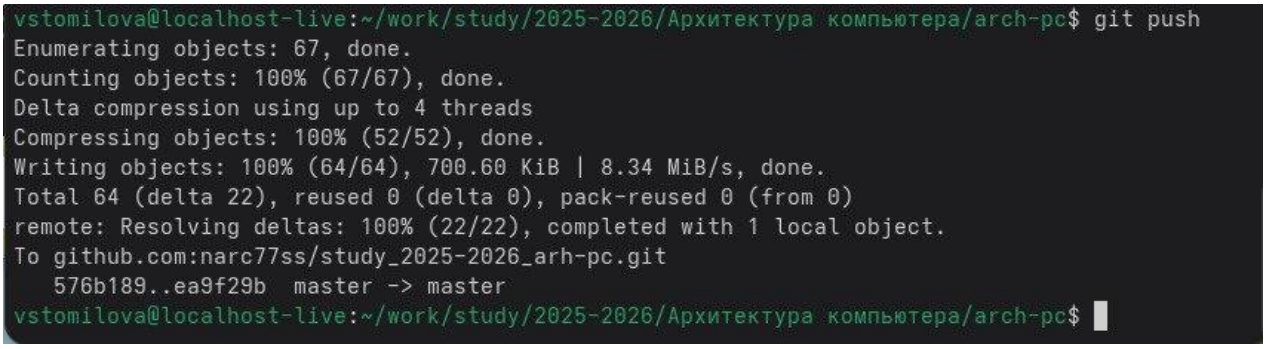


Рисунок 10. Сжатие файлов

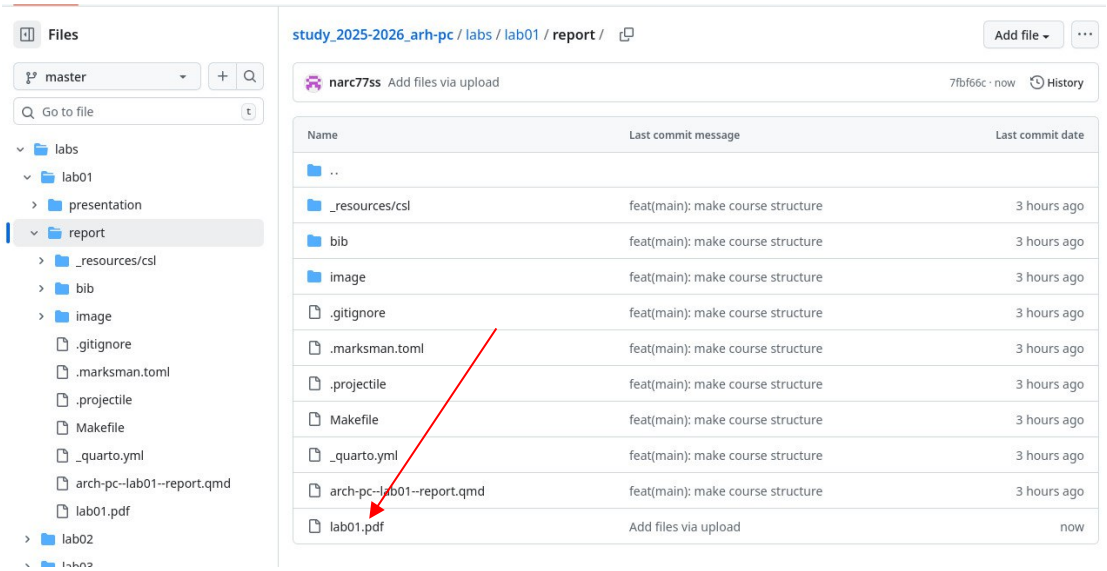
Я завершила настройку структуры курса, а затем использовала основной рабочий цикл Git: добавила изменения в индекс (**git add**), зафиксировала их с осмысленным комментарием (**git commit**) и отправила на удаленный сервер (**git push**). Теперь все мои локальные изменения синхронизированы с **GitHub**.

Выполнение самостоятельной работы

Задание №1. Создайте отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства (labs/lab02/report).

Задание №2. Скопируйте отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги созданного рабочего пространства.

Задание №3. Загрузите файлы на github.



Выводы

В ходе выполнения данной лабораторной работы я изучила теоретические основы и получила практические навыки работы с системой контроля версий **Git**. Я научилась выполнять базовую настройку **Git**, создавать и настраивать репозитории на **GitHub**, использовать **SSH-ключи** для безопасного соединения, а также освоила основной рабочий цикл: добавление изменений (add), их фиксацию (commit) и отправку на удаленный сервер (push). Цель работы была полностью достигнута.

1. Введение - О системе контроля версий/ https://git-scm.com/book/ru/v2/%D0%92%D0%B2%D0%B5%D0%B4%D0%B5%D0%BD%D0%B8%D0%B5-%D0%9E-%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B5-%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8F-%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B9
2. What is Git version control? / https://about.gitlab.com/topics/version-control/what-is-git-version-control/
3. Основные команды GIT / https://habr.com/ru/articles/918386/