

# 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: Валиева Карина Ринатовна

Группа: НБИбд-01-25

МОСКВА

2025 г.

## Оглавление

1 Цель работы.....	3
2 Задание.....	4
3 Теоретическое введение.....	5
4 Выполнение лабораторной работы.....	7
4.1 Настройка GitHub .....	7
4.2 Базовая настройка Git.....	7
4.3 Создание SSH – ключа .....	8
4.4 Создание рабочего пространства и репозитория курса на основе шаблона.....	9
4.5 Создание репозитория курса на основе шаблона.....	10
4.6 Настройка каталога курса .....	11
4.7 Выполнение заданий для самостоятельной работы.....	12
5 Выводы.....	15

## **1 Цель работы**

Целью данной работы является изучить идеологию и применение средств контроля версий, а также 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

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 изменения, сделанные разными участниками, 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

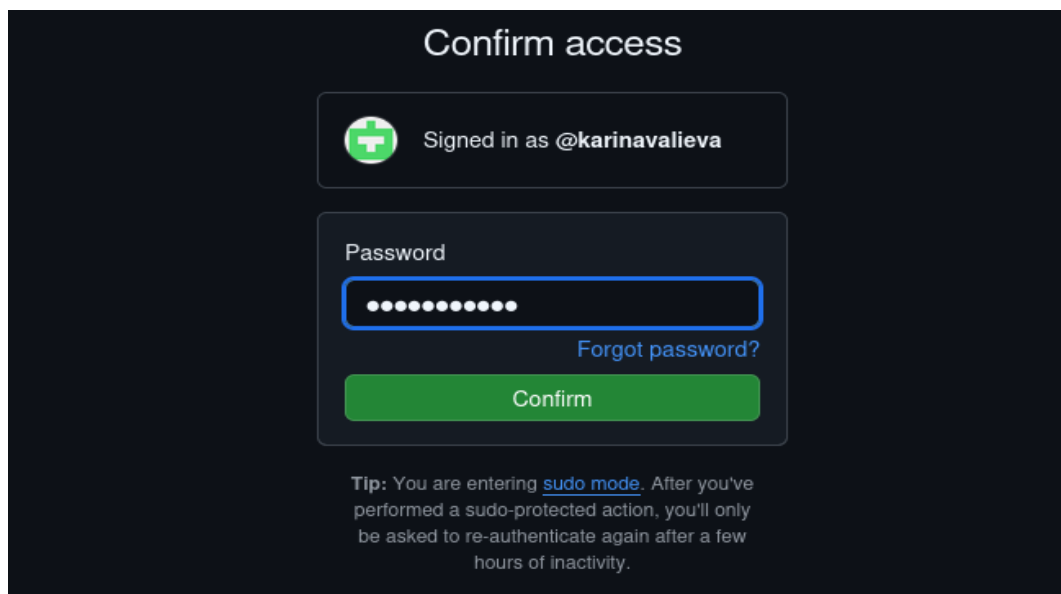
точки ветвления версий и собственные истории изменений каждой ветви. Обычно доступна информация о том, кто из участников, когда и какие изменения вносил. Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить.

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

Создаю учетную запись на сайте GitHub. Далее я заполнила основные данные учетной записи (Рис.1.).



Confirm access

Signed in as @karinavalieva

Password

.....

[Forgot password?](#)

Confirm

**Tip:** You are entering [sudo mode](#). After you've performed a sudo-protected action, you'll only be asked to re-authenticate again after a few hours of inactivity.

Рис. 1. Заполнение данных учетной записи GitHub

### 4.2 Базовая настройка Git

Открываю виртуальную машину, затем открываю терминал и делаю предварительную конфигурацию git. Ввожу команду `git config --global user.name ""`, указывая свое имя и команду `git config --global user.email "work@mail"`, указывая в ней электронную почту владельца. (Рис.2.)

```
krvalieva@vbox:~$ git config --global user.name "<Karina Valieva>"
krvalieva@vbox:~$ git config --global user.email "<karina.valieva.2007@mail.ru>"
```

Рис. 2. Предварительная конфигурация git

Настраиваю utf-8 в выводе сообщений git для корректного отображения символов. (Рис.3.)

```
krvalieva@vbox:~$ git config --global core.quotePath false
```

Рис. 3. Настройка кодировки

Задаю имя «master» для начальной ветки.(Рис.4.)

```
krvalieva@vbox:~$ git config --global init.defaultBranch master
```

Рис. 4. Создание имени для начальной ветки

Задаю параметр autocrlf со значением input. CR и LF – это символы, которые можно использовать для обозначения разрыва строки в текстовых файлах. (Рис.5.)

```
krvalieva@vbox:~$ git config --global core.autocrlf input
```

Рис. 5. Параметр autocrlf

Задаю параметр safecrlf со значением warn, так Git будет проверять преобразование на обратимость. (Рис.6.)

```
krvalieva@vbox:~$ git config --global core.safecrlf warn
```

Рис. 6. Параметр safecrlf

### 4.3 Создание SSH – ключа

Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать приватный и открытый ключ. Ввожу команду ssh-keygen -C «Имя Фамилия, work@email», указывая имя и электронную почту владельца. (Рис.7.)

```
krvalieva@vbox:~$ ssh-keygen -C "Karina Valieva <karina.valieva.2007@mail.ru>"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/krvalieva/.ssh/id_ed25519):
Enter passphrase for "/home/krvalieva/.ssh/id_ed25519" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/krvalieva/.ssh/id_ed25519
Your public key has been saved in /home/krvalieva/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:nUjFQaS+RQ3L3uBRopa0uXSiPa2mIynbtdcyRVRE0ZQ Karina Valieva <karina.valieva.2007@mail.ru>
The key's randomart image is:
+--[ED25519 256]--+
|      .+B**o      |
|      . 0o*E      |
|      @.B ..      |
|      B @.=       |
|      . S.B .     |
|      =.         |
|      .. +o      |
|      ..0..++ .   |
|      .o..oo o    |
+-----[SHA256]-----+
```

Рис. 7. Генерация SSH – ключа

Копирую открытый ключ из директории, в которой он был сохранен, с помощью утилиты xclip. (Рис.8.)

```
krvalieva@vbox:~$ cat ~/.ssh/id_ed25519.pub | xclip -sel clip
```

Рис. 8. Копирование содержимого файла

Открываю браузер, захожу на сайт GitHub. Открываю свой профиль и выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key» (Рис.9.)



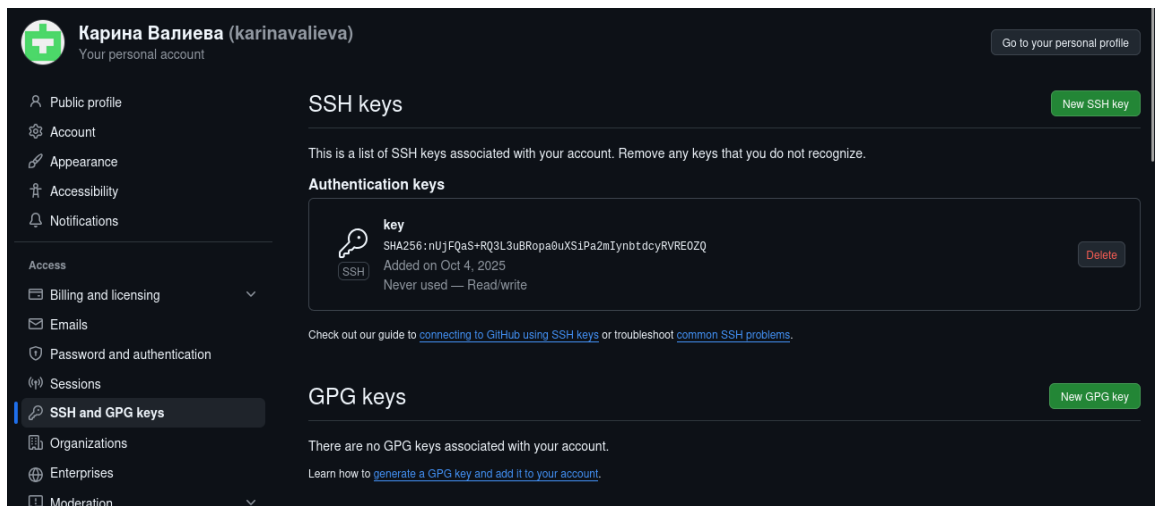


Рис. 9. Окно SSH and GPG keys

Вставляю скопированный ключ в поле «Key». В поле Title указываю имя для ключа. Нажимаю «Add SSH-key», чтобы завершить добавление ключа.(Рис.10.)

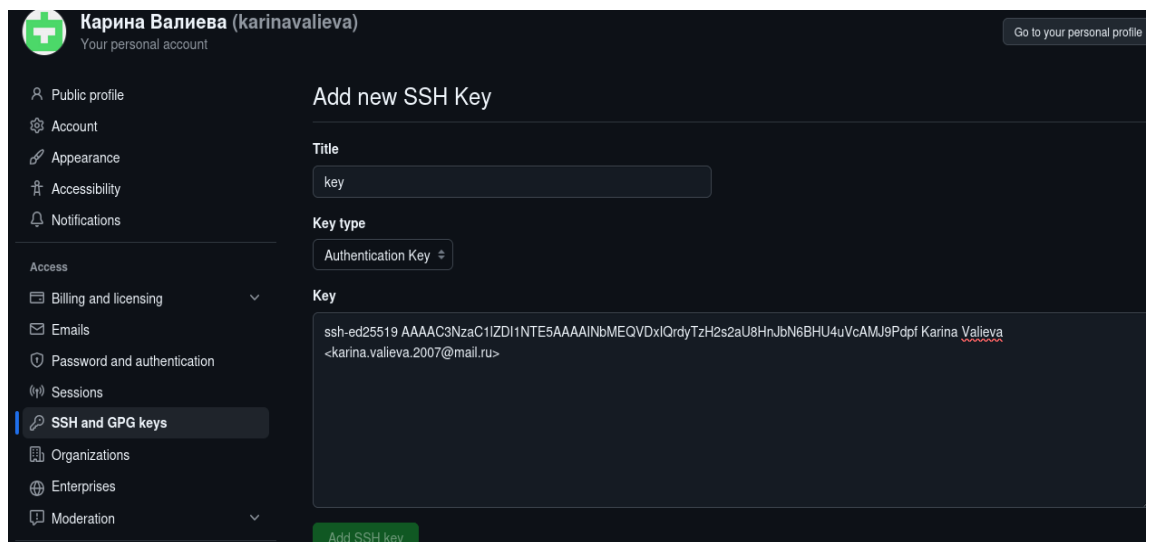


Рис. 10. Добавление ключа

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

Открываю терминал и создаю директорию, рабочее пространство, с помощью утилиты mkdir, благодаря ключу -p создаю все директории после домашней ~/work/study/2025-2026/“Архитектура компьютера” рекурсивно. Далее проверяю с помощью ls, действительно ли были созданы необходимые мне каталоги. (Рис.11.)

```
krvalieva@vbox:~$ mkdir -p ~/work/study/2025-2026/"Архитектура компьютера"
krvalieva@vbox:~$ ls
Desktop      Music        Public        Templates
Documents    parentdir    'sudo upt update'  Videos
Downloads    Pictures     'sudo upt update.pub' work
```

Рис. 11.Создание рабочего пространства

#### 4.5 Создание репозитория курса на основе шаблона

В браузере перехожу на страницу репозитория с шаблоном курса по адресу <https://github.com/yamadharma/course-directory-student-template>. Далее выбираю «Use this template», чтобы использовать этот шаблон для своего репозитория. В открывшемся окне задаю имя репозитория (Repository name): study\_2025–2026\_arh-рс и создаю репозиторий, нажимаю на кнопку «Create repository» (Рис.12.)

Рис. 12.Окно создания репозитория

Созданный репозиторий. (Рис.13.)

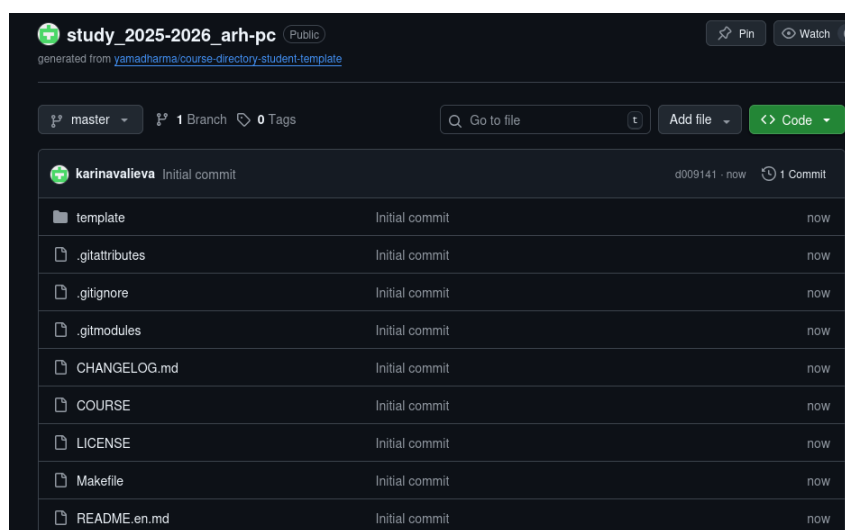


Рис. 13.Созданный репозиторий

Через терминал перехожу в созданный каталог курса с помощью утилиты `cd`. Клонировую созданный репозиторий с помощью команды `git clone --recursive git@github.com:/study_2025–2026_arh-pc.git arch-pc`. (Рис.14.)

```
krvalieva@vbox:~$ cd ~/work/study/2025-2026/"Архитектура компьютера"
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера$ git clone --recursive git@github.com:karinavalieva/study_2025-2026_arh-pc.git arch-pc
Cloning into 'arch-pc'...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
```

Рис. 14. Клонирование репозитория

#### 4.6 Настройка каталога курса

Перехожу в каталог `arch-pc` с помощью утилиты `cd`. (Рис.15.)

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера$ cd arch-pc
```

Рис. 15. Перемещение между директориями

Создаю необходимые каталоги. (Рис.16.)

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc$ echo arch-pc > COURSE
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc$ make prepare
```

Рис. 16. Создание каталогов

Отправляю созданные каталоги с локального репозитория на сервер: добавляю все созданные каталоги с помощью `git add`, комментирую и сохраняю изменения на сервере как добавление курса с помощью `git commit`. (Рис.17.)

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc$ git add .
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc$ git commit -am 'feat(main): make course structure'
[master 129573a] feat(main): make course structure
260 files changed, 8746 insertions(+), 216 deletions(-)
delete mode 100644 CHANGELOG.md
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/.gitignore
create mode 100644 labs/lab01/presentation/.marksman.toml
create mode 100644 labs/lab01/presentation/.projectile
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/_assets/auto/beamer.el
create mode 100644 labs/lab01/presentation/_assets/auto/preamble.el
create mode 100644 labs/lab01/presentation/_assets/beamer.tex
create mode 100644 labs/lab01/presentation/_quarto.yml
create mode 100644 labs/lab01/presentation/_resources/image/logo_rudn.png
create mode 100644 labs/lab01/presentation/arch-pc--lab01--presentation.qmd
create mode 100644 labs/lab01/presentation/image/kulyabov.jpg
create mode 100644 labs/lab01/report/.gitignore
create mode 100644 labs/lab01/report/.marksman.toml
create mode 100644 labs/lab01/report/.projectile
create mode 100644 labs/lab01/report/Makefile
create mode 100644 labs/lab01/report/_assets/preamble.tex
create mode 100644 labs/lab01/report/_quarto.yml
create mode 100644 labs/lab01/report/_resources/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab01/report/arch-pc--lab01--report.qmd
create mode 100644 labs/lab01/report/bib/cite.bib
create mode 100644 labs/lab01/report/image/solvay.jpg
create mode 100644 labs/lab02/presentation/.gitignore
create mode 100644 labs/lab02/presentation/.marksman.toml
create mode 100644 labs/lab02/presentation/.projectile
create mode 100644 labs/lab02/presentation/Makefile
create mode 100644 labs/lab02/presentation/_assets/auto/beamer.el
create mode 100644 labs/lab02/presentation/_assets/auto/preamble.el
```

Рис. 17. Добавление и сохранение изменений на сервере

Отправляю все на сервер с помощью push. (Рис.18.)

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc$ git push
Enumerating objects: 74, done.
Counting objects: 100% (74/74), done.
Compressing objects: 100% (58/58), done.
Writing objects: 100% (71/71), 700.99 KiB | 5.52 MiB/s, done.
Total 71 (delta 25), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (25/25), completed with 1 local object.
```

Рис. 18. Выгрузка изменений на сервер

Проверяю правильность выполнения работы на сайте GitHub. (Рис.19.)

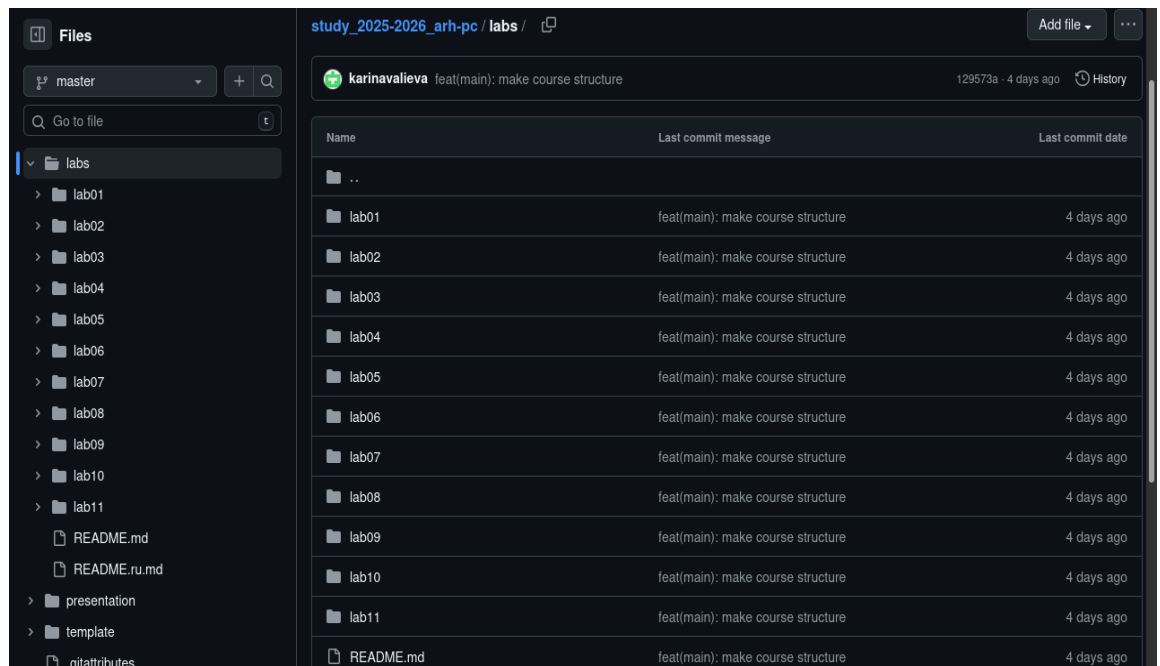


Рис. 19.Страница репозитория

#### 4.7 Выполнение заданий для самостоятельной работы

1. С помощью утилиты cd перехожу в labs/lab02/report (Рис.20.) и создаю файл для отчета по этой лабораторной работе. (Рис.21.)

```
krvalieva@vbox:~$ cd ~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab02/report
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab02/report$
```

Рис. 20. Перемещение между директориями

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab02/report$ touch Л02_Валиева_отчет
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab02/report$
```

Рис. 21. Создание файла

2. Перемещаюсь в каталог предыдущей лабораторной работы с помощью cd. (Рис.22.)

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab02/report$ cd ..
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab02$ cd ..
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs$ cd lab01/report
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab01/report$
```

Рис. 22. Перемещение между директориями

Проверяю расположение файлы с отчетом по лабораторной работе №1.  
(Рис.23.)

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab01/report$ ls ~/Downloads
tsetup.6.1.3  tsetup.6.1.3.tar.xz  Л01_Валиева_отчет.pdf
```

Рис. 23. Проверка расположения нужного файла

Копирую этот файл в lab01/report и проверяю выполнение с помощью утилиты ls. (Рис.24.)

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab01/report$ cp ~/Downloads/Л01_Валиева_отчет.pdf ~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab01/report
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab01/report$ ls
arch-pc--lab01--report.qmd  _assets  bib  image  Makefile  _quarto.yml  _resources  Л01_Валиева_отчет.pdf
```

Рис. 24. Копирование файла и проверка выполнения команды

3. Добавляю файл Л01\_Валиева\_отчет.pdf на сервер и сохраняю изменения командой `git commit -m "..."`, указывая, какое именно изменение произошло.  
(Рис.25.)

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab01/report$ git add Л01_Валиева_отчет.pdf
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab01/report$ git commit -m "Add existing file"
[master 76d0fc3] Add existing file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 labs/lab01/report/Л01_Валиева_отчет.pdf
```

Рис. 25. Добавление файла и сохранение изменений

Отправляю в центральный репозиторий сохраненные изменения. (Рис.26.)

```
krvalieva@vbox:~/work/study/2025-2026/Архитектура компьютера/arch-pc/labs/lab01/report$ git push -f origin master
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 980.07 KiB | 6.85 MiB/s, done.
Total 6 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:karinavalieva/study_2025-2026_arh-pc.git
129573a..76d0fc3  master -> master
```

Рис. 26. Отправка изменений в центральный репозиторий

Проверяю правильность выполненных действий на сайте. На (Рис.27.) смотрю, добавлен ли мой комментарий, а на (Рис.28.) отображается ли добавленный файл.

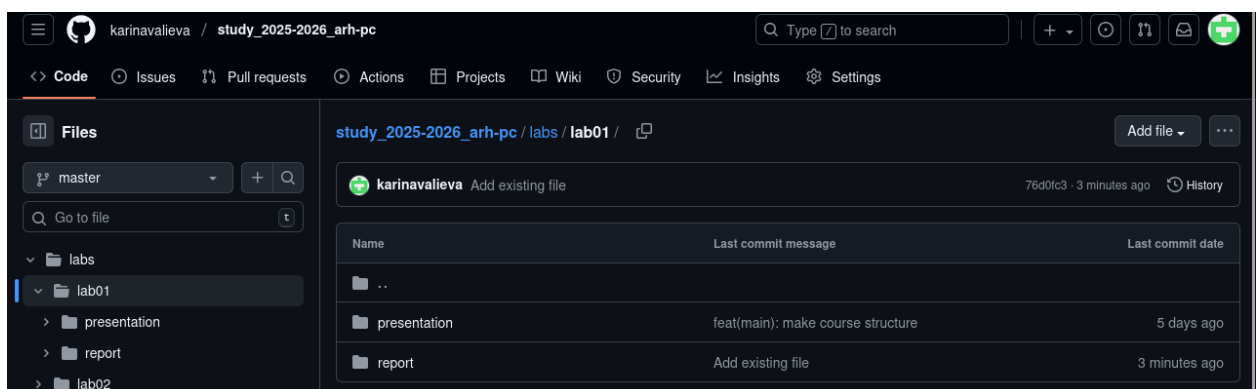
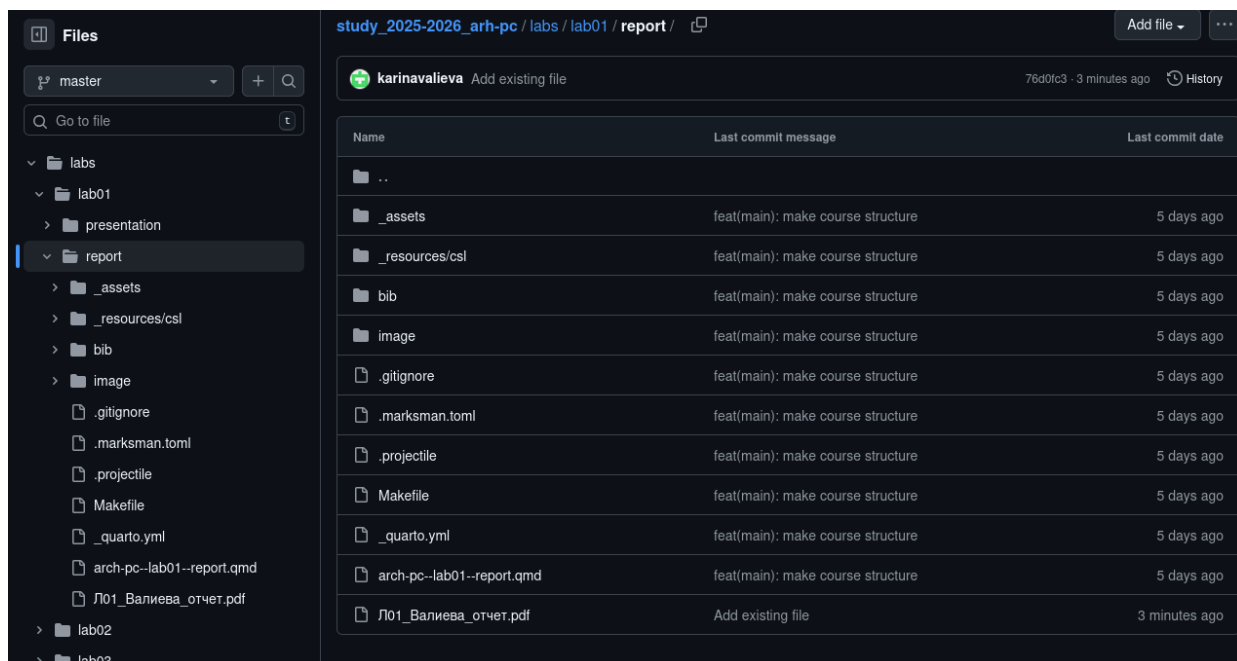


Рис. 27. Отображение комментария



**Рис. 28. Проверка наличия файла**

После создания отчета по данной лабораторной работе я выполняю аналогичные действия пункту 2 и 3 для самостоятельной работы.

## **5 Выводы**

При выполнении данной лабораторной работы я изучила идеологию и применение средств контроля версий, а также приобрела практические навыки по работе с системой git.