

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: Довашеев Дмитрий

Группа: НКАбд-07-25

МОСКВА

2025 г.

Оглавление

1 Цель работы.....	3
3 Теоретическое введение	5
4 Выполнение лабораторной работы.....	7
4.1 Техническое обеспечение	7
4.2 Базовая настройка Git	7
4.3 Создание рабочего пространства и репозитория курса на основе шаблона.....	10
4.4 Создание репозитория курса на основе шаблона.	10
4.5 Настройка каталога курса.....	13
5 Задания для самостоятельной работы.....	14
6 Выводы.....	15
Список литературы	16

1 Цель работы

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

2 Задание

На основе методических указаний провести работу с базовыми командами системы контроля версий git, выучить применение команд для разных случаев использования, настроить GitHub.

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.

<code>git commit -am</code> 'Описание коммита'	сохранить все добавленные изменения и все изменённые файлы
<code>git checkout -b</code> имя_ветки	создание новой ветки, базирующейся на текущей
<code>git checkout</code> имя_ветки	переключение на некоторую ветку (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой)
<code>git push origin</code> имя_ветки	отправка изменений конкретной ветки в центральный репозиторий
<code>git merge</code> --no-ff имя_ветки	слияние ветки с текущим деревом
<code>git branch -d</code> имя_ветки	удаление локальной уже слитой с основным деревом ветки
<code>git branch -D</code> имя_ветки	принудительное удаление локальной ветки
<code>git push origin</code> :имя_ветки	удаление ветки с центрального репозитория

Таблица 3.1 Описание некоторых команд системы контроля версий Git.

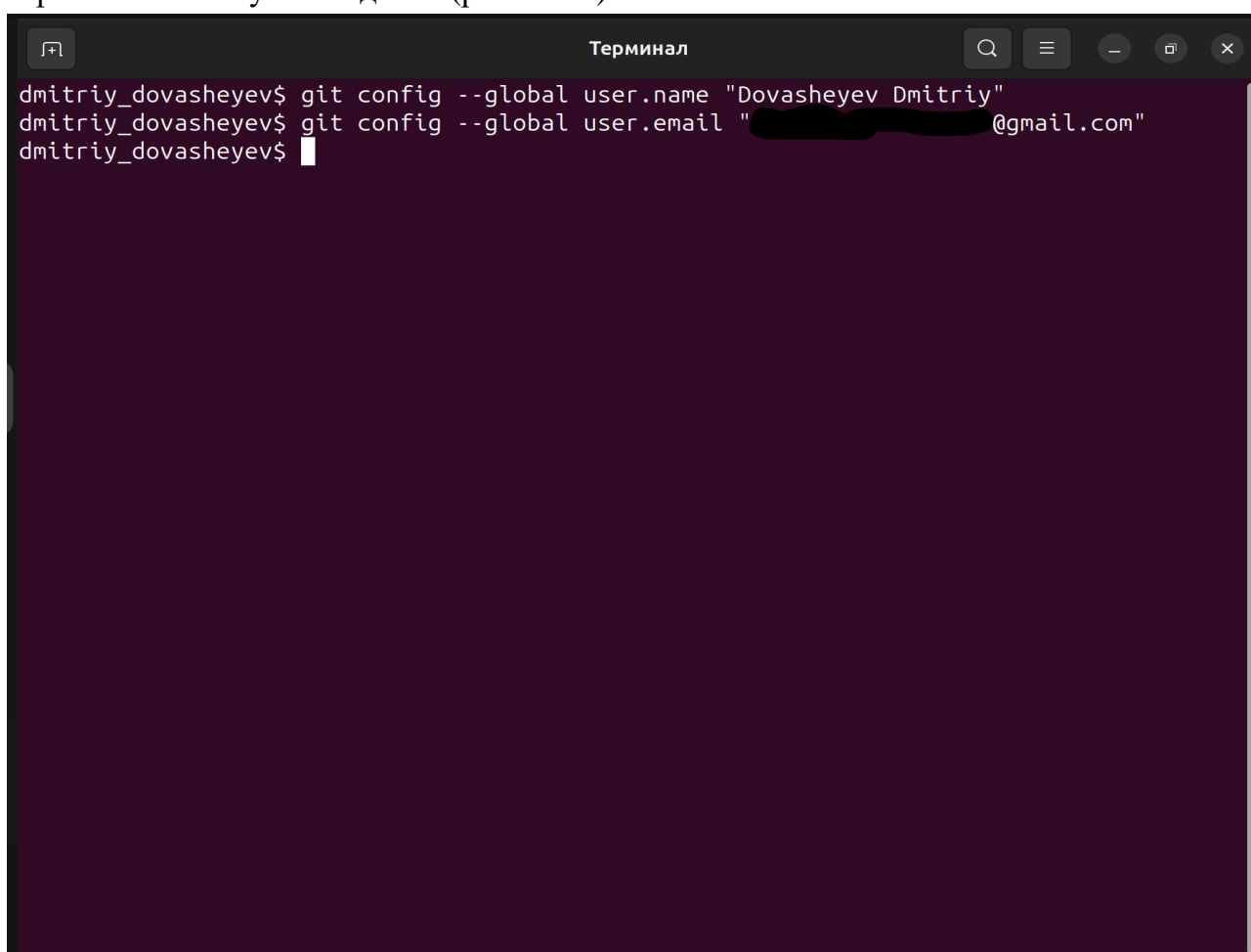
4 Выполнение лабораторной работы

4.1 Техническое обеспечение

Лабораторная работа была выполнена на домашнем компьютере под управлением операционной системы Ubuntu 24.04.3 LTS.

4.2 Базовая настройка Git

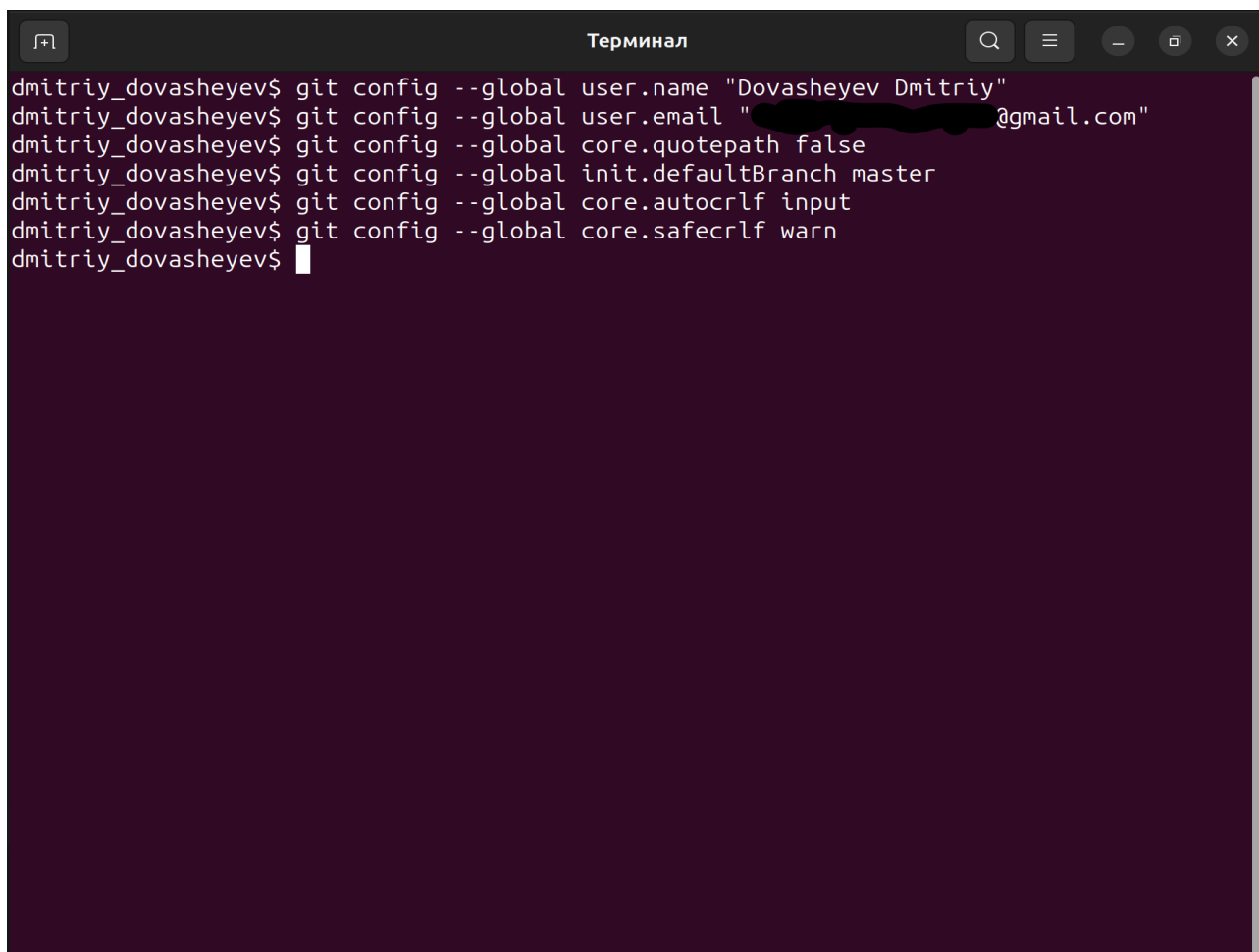
Для начала я проведу предварительную конфигурацию Git, для этого открываю терминал и ввожу команды на (рис. 4.2.1)



```
Терминал
dmitriy_dovasheyev$ git config --global user.name "Dovasheyev Dmitriy"
dmitriy_dovasheyev$ git config --global user.email "[REDACTED]@gmail.com"
dmitriy_dovasheyev$
```

Рис. 4.2.1 Предварительная конфигурация Git.

Далее настраиваю параметры utf-8, имя начальной ветки, autocrlf и safecrlf (рис. 4.2.2)

A terminal window titled "Терминал" (Terminal) with a dark background. It shows a series of six Git configuration commands being executed by a user named "dmitriy_dovasheyev". The commands are: 1. "git config --global user.name 'Dovasheyev Dmitriy'", 2. "git config --global user.email '[redacted]@gmail.com'", 3. "git config --global core.quotePath false", 4. "git config --global init.defaultBranch master", 5. "git config --global core.autocrlf input", and 6. "git config --global core.safecrlf warn". Each command is followed by a prompt character "\$" and a new line. The terminal has standard window controls (minimize, maximize, close) and a search icon in the top right corner.

```
dmitriy_dovasheyev$ git config --global user.name "Dovasheyev Dmitriy"
dmitriy_dovasheyev$ git config --global user.email "[redacted]@gmail.com"
dmitriy_dovasheyev$ git config --global core.quotePath false
dmitriy_dovasheyev$ git config --global init.defaultBranch master
dmitriy_dovasheyev$ git config --global core.autocrlf input
dmitriy_dovasheyev$ git config --global core.safecrlf warn
dmitriy_dovasheyev$
```

Рис. 4.2.2 Настройка параметров Git.

Далее создаю пару ssh ключей для интеграции с платформой GitHub (рис. 4.2.3)


```
Терминал
dmitriy_dovasheyev$ ssh-keygen -C "Dovasheyev Dmitriy <dmitriydovasheyev@gmail.com>"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/dmitriy_dovasheyev/.ssh/id_ed25519):
Created directory '/home/dmitriy_dovasheyev/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Passphrases do not match. Try again.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/dmitriy_dovasheyev/.ssh/id_ed25519
Your public key has been saved in /home/dmitriy_dovasheyev/.ssh/id_ed25519.pub
The key fingerprint is:
[redacted] Dovasheyev Dmitriy [redacted]
The key's randomart image is:
[redacted]
```

Рис. 4.2.3 Создание пары ssh ключей.

Далее я перехожу на сайт GitHub, авторизуюсь, перехожу в настройки аккаунта, вставляю публичный ключ в предназначенном для этого поле. (рис. 4.2.4)

Add new SSH Key

Title
ubuntu

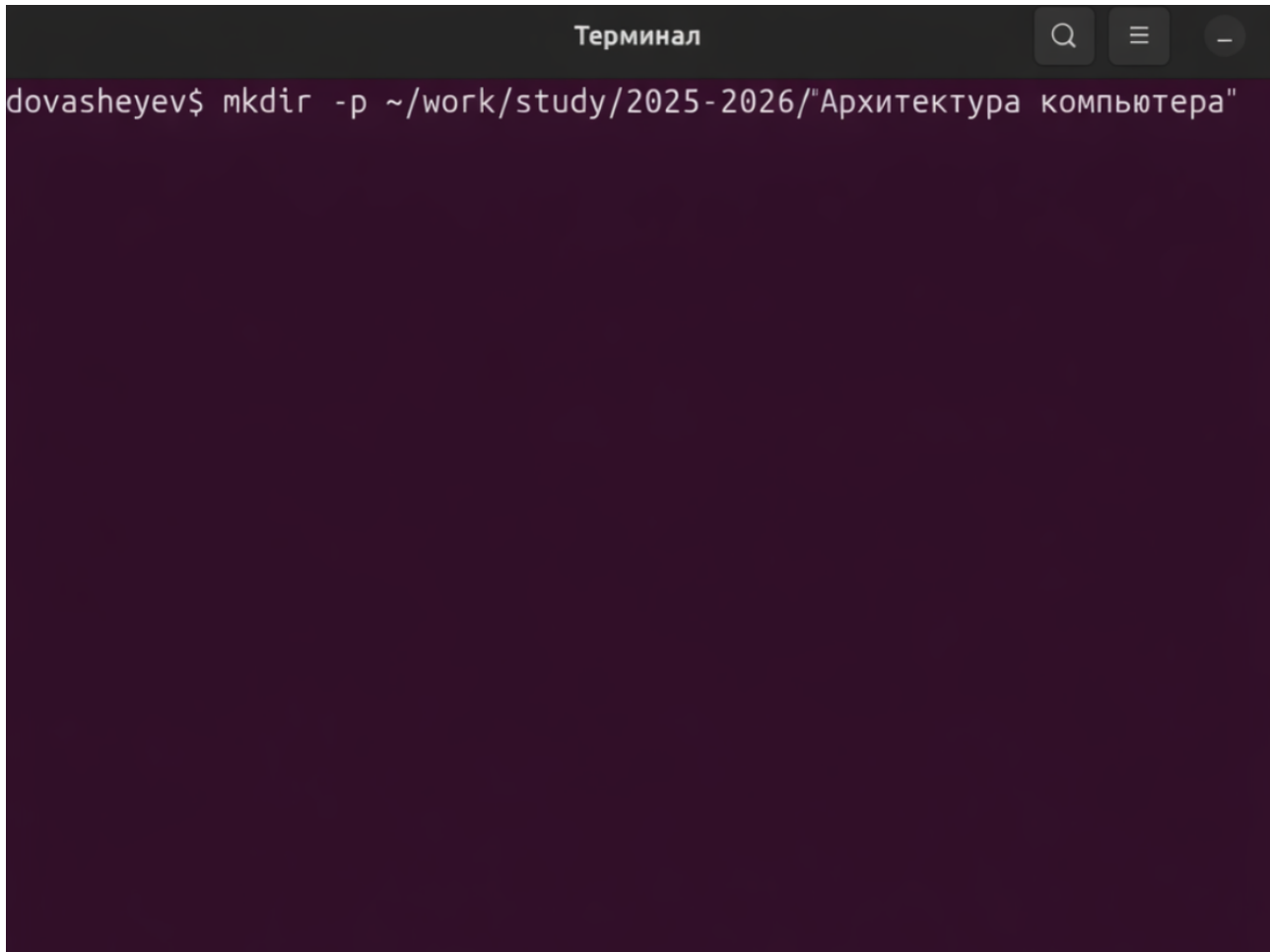
Key type
Authentication Key

Key
Dovasheyev Dmitriy [redacted]

Add SSH key

4.3 Создание рабочего пространства и репозитория курса на основе шаблона.

Рабочее пространство при выполнении лабораторных работ должно придерживаться определённой структурной иерархии, для этого я создаю директорию на своем рабочем компьютере (рис. 4.3.1)



```
Терминал
dovasheev$ mkdir -p ~/work/study/2025-2026/"Архитектура компьютера"
```

4.4 Создание репозитория курса на основе шаблона.

Создаю репозиторий на основе имеющего шаблона (рис. 4.4.1) через функционал клонирования интерфейса GitHub. (рис 4.4.2)

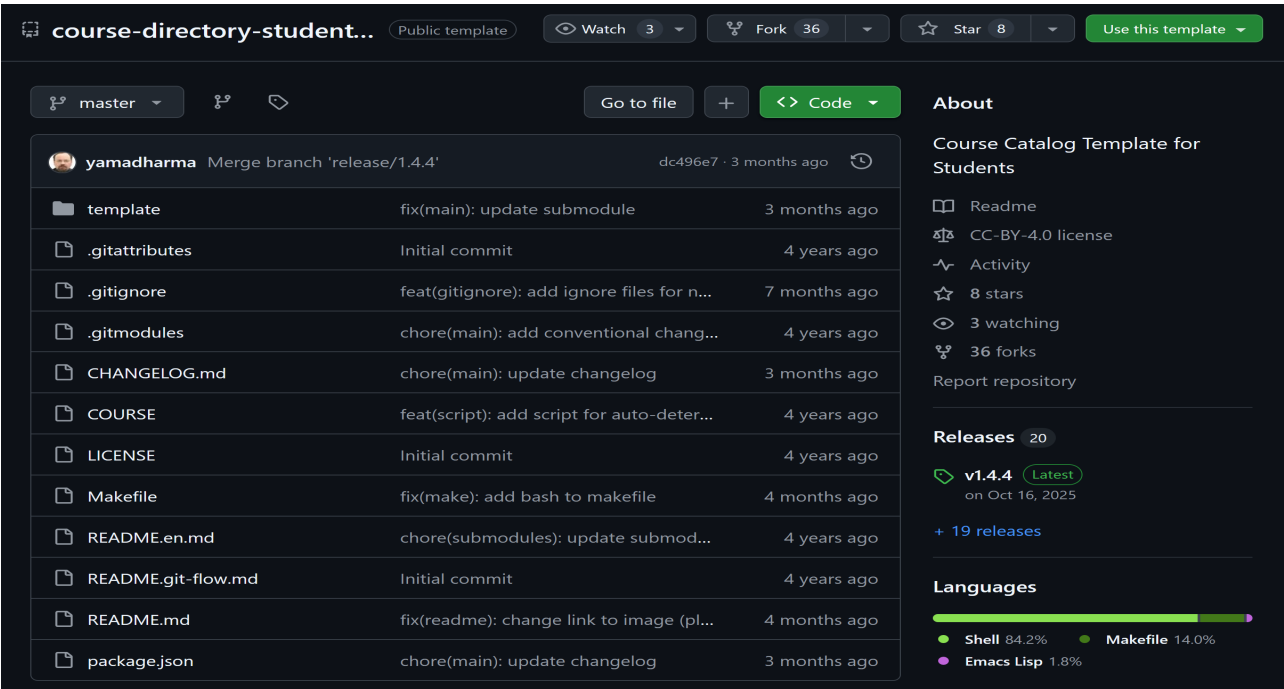


Рис. 4.4.1 Шаблон для клонирования на платформе GitHub.

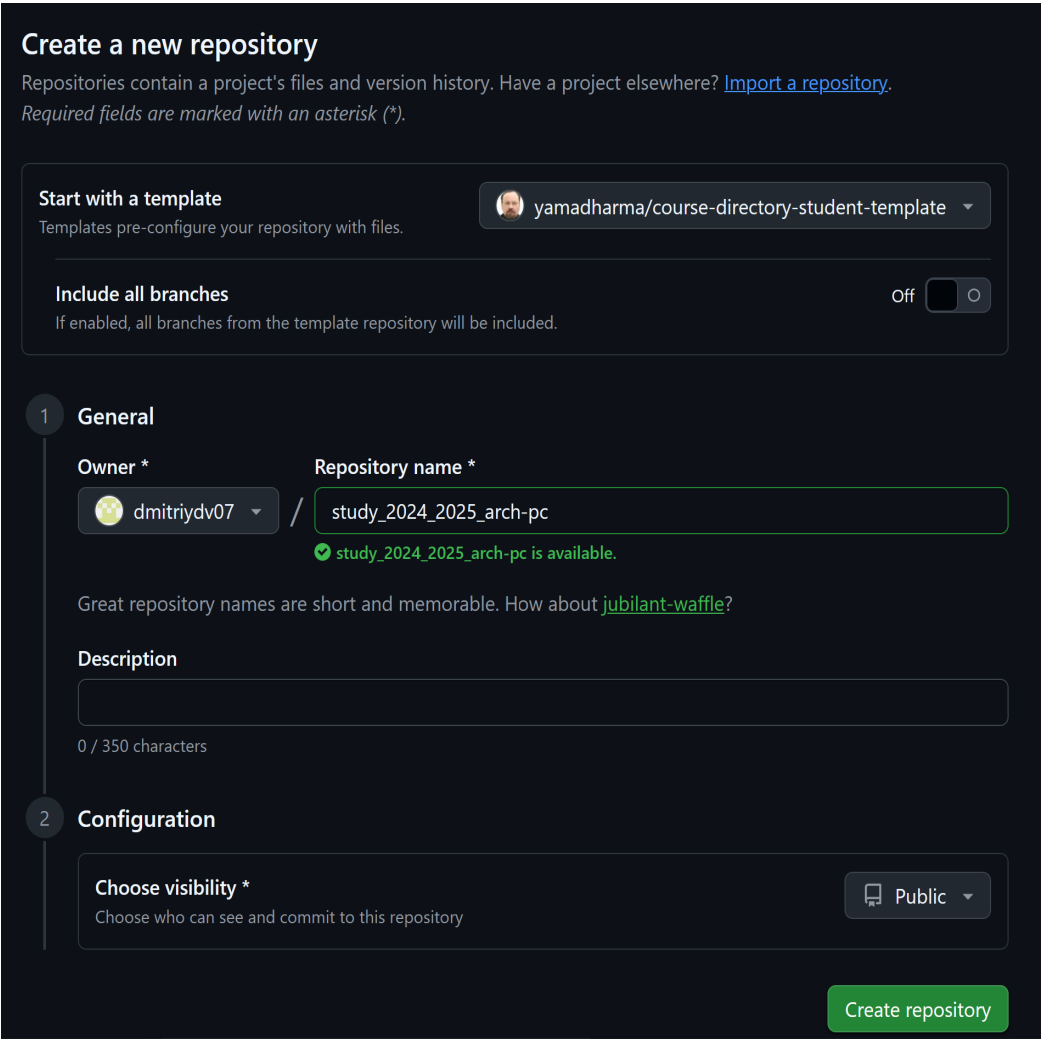


Рис. 4.4.2 Процесс клонирования репозитория через интерфейс GitHub.

Сгенерированный репозиторий на основе шаблона клонирую на свой рабочий компьютер, для этого беру ссылку для клонирования через интерфейс GitHub (рис. 4.4.3) и затем ввожу в терминале `git clone`. (рис 4.4.4)

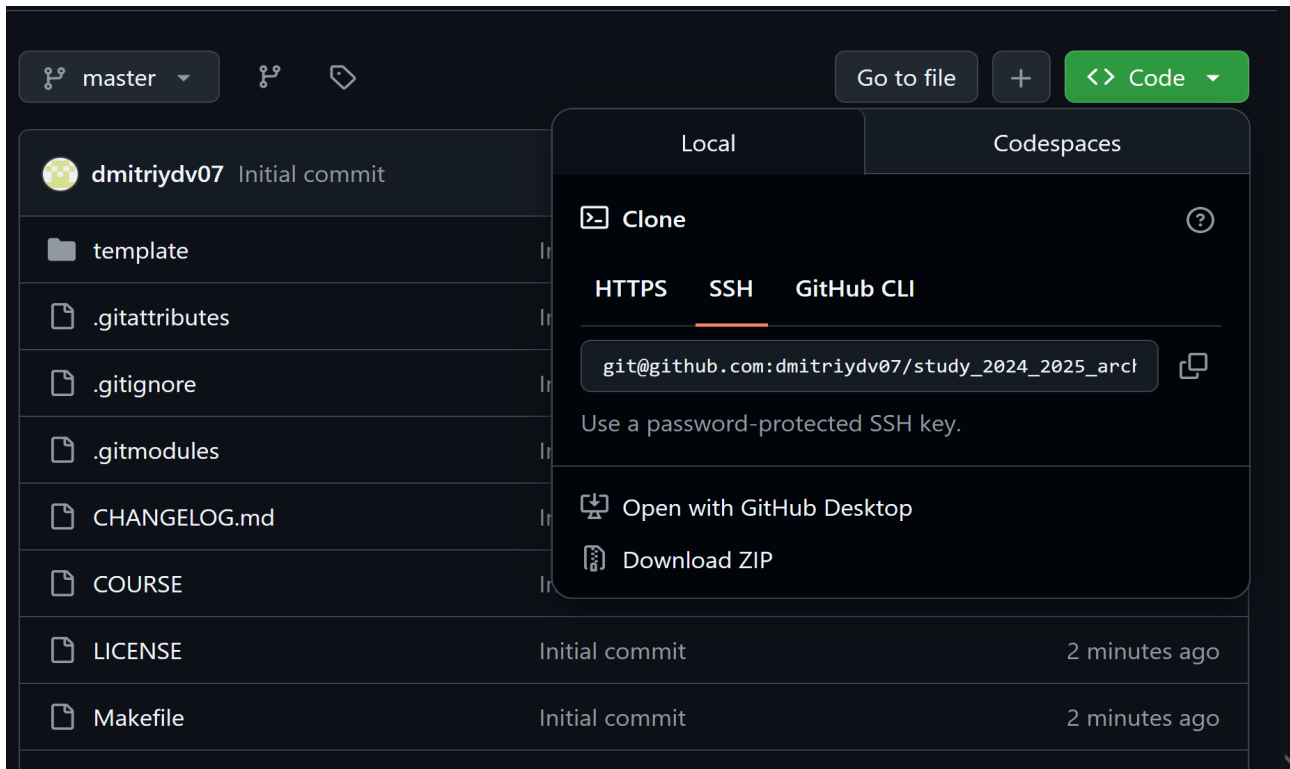


Рис. 4.4.3 Копирование ссылки для последующей вставки в терминал.

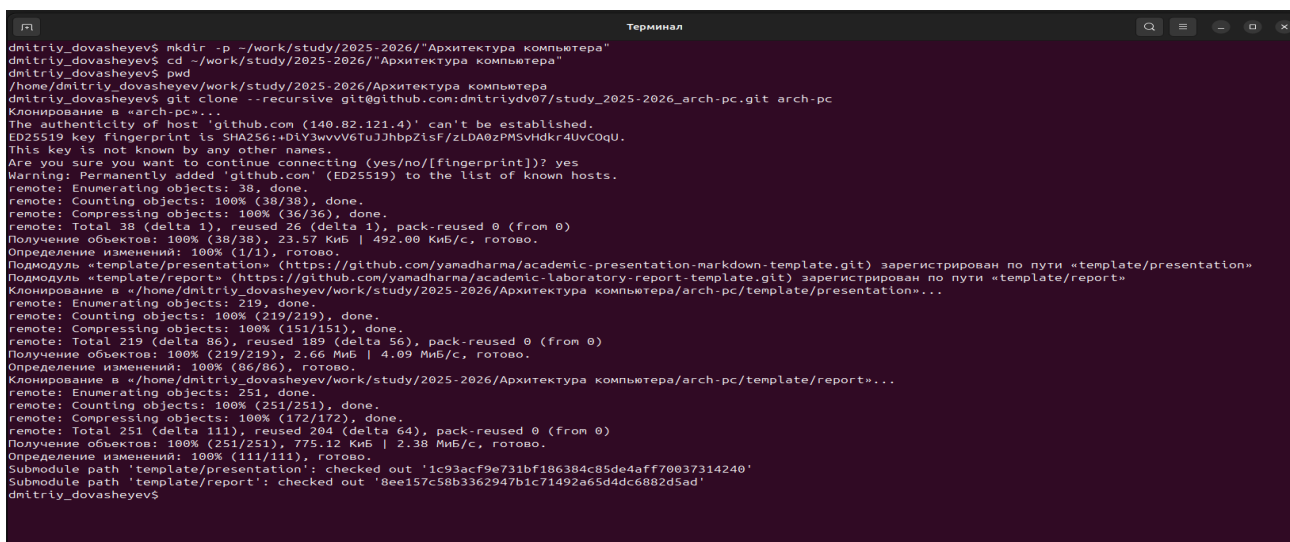
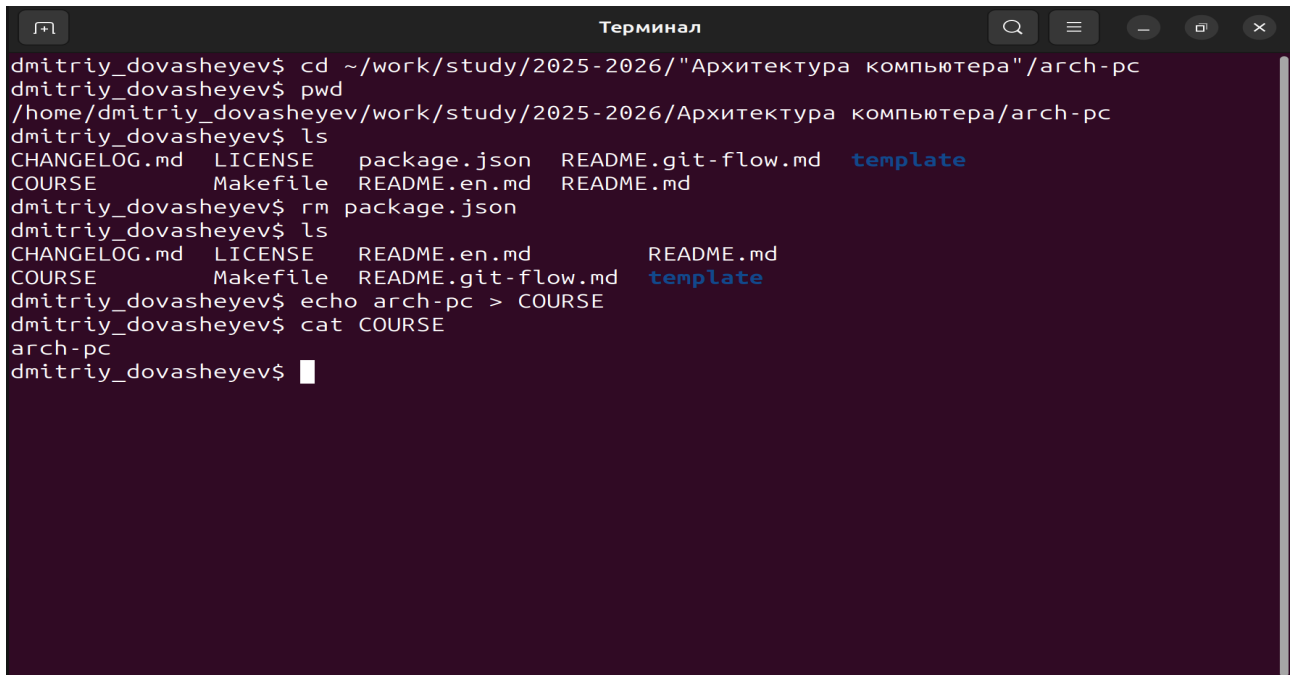


Рис. 4.4.4 Копирование репозитория на рабочий компьютер.

4.5 Настройка каталога курса.

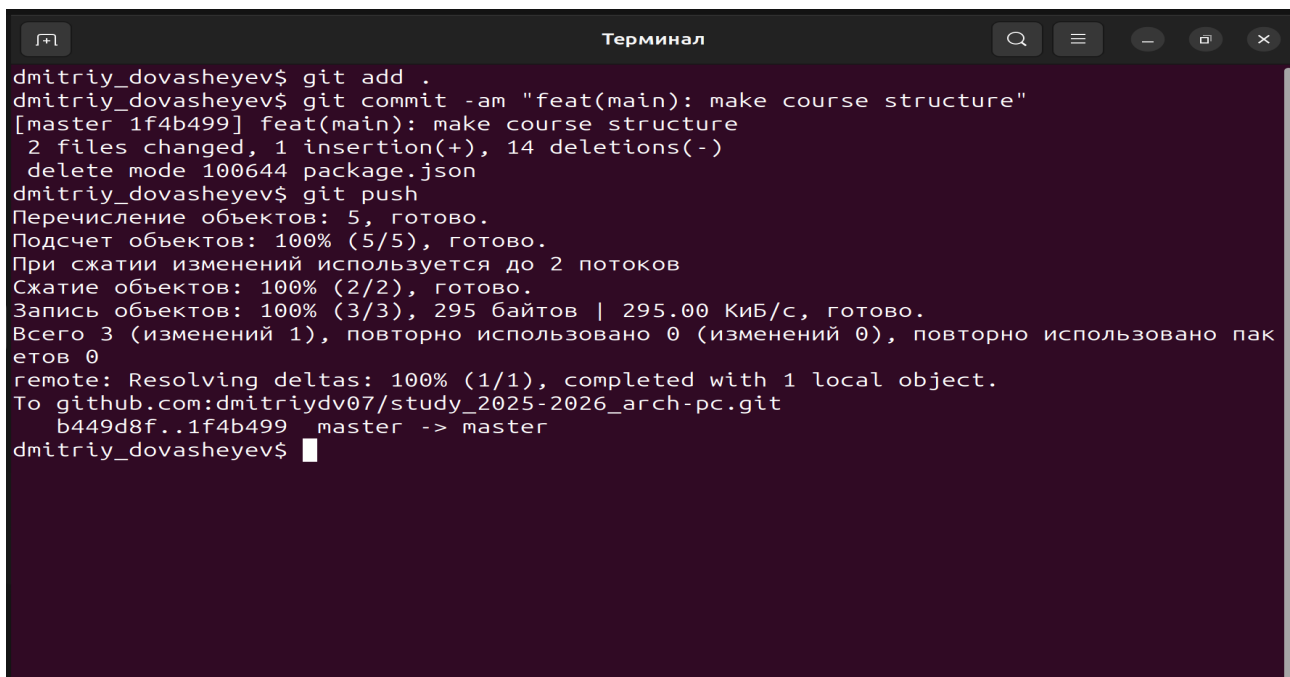
В каталоге курса удаляю лишние файлы и формирую необходимые каталоги (рис. 4.3.1)

A terminal window titled "Терминал" with a dark background. The user, dmitriy_dovasheyev, navigates to the directory ~/work/study/2025-2026/"Архитектура компьютера"/arch-pc. They list the current files: CHANGELOG.md, LICENSE, package.json, README.git-flow.md, template, COURSE, Makefile, README.en.md, and README.md. Then, they remove package.json. They list the files again, showing the same set except for package.json. Finally, they create a new directory named COURSE using the command echo arch-pc > COURSE and verify its creation with cat COURSE.

```
dmitriy_dovasheyev$ cd ~/work/study/2025-2026/"Архитектура компьютера"/arch-pc
dmitriy_dovasheyev$ pwd
/home/dmitriy_dovasheyev/work/study/2025-2026/Архитектура компьютера/arch-pc
dmitriy_dovasheyev$ ls
CHANGELOG.md  LICENSE      package.json  README.git-flow.md  template
COURSE        Makefile     README.en.md  README.md
dmitriy_dovasheyev$ rm package.json
dmitriy_dovasheyev$ ls
CHANGELOG.md  LICENSE      README.en.md  README.md
COURSE        Makefile     README.git-flow.md  template
dmitriy_dovasheyev$ echo arch-pc > COURSE
dmitriy_dovasheyev$ cat COURSE
arch-pc
dmitriy_dovasheyev$
```

Рис. 4.3.1 Настройка каталога курса.

Делаю снимок сделанных изменений и push'у их на свой репозиторий в GitHub. (рис 4.3.2)

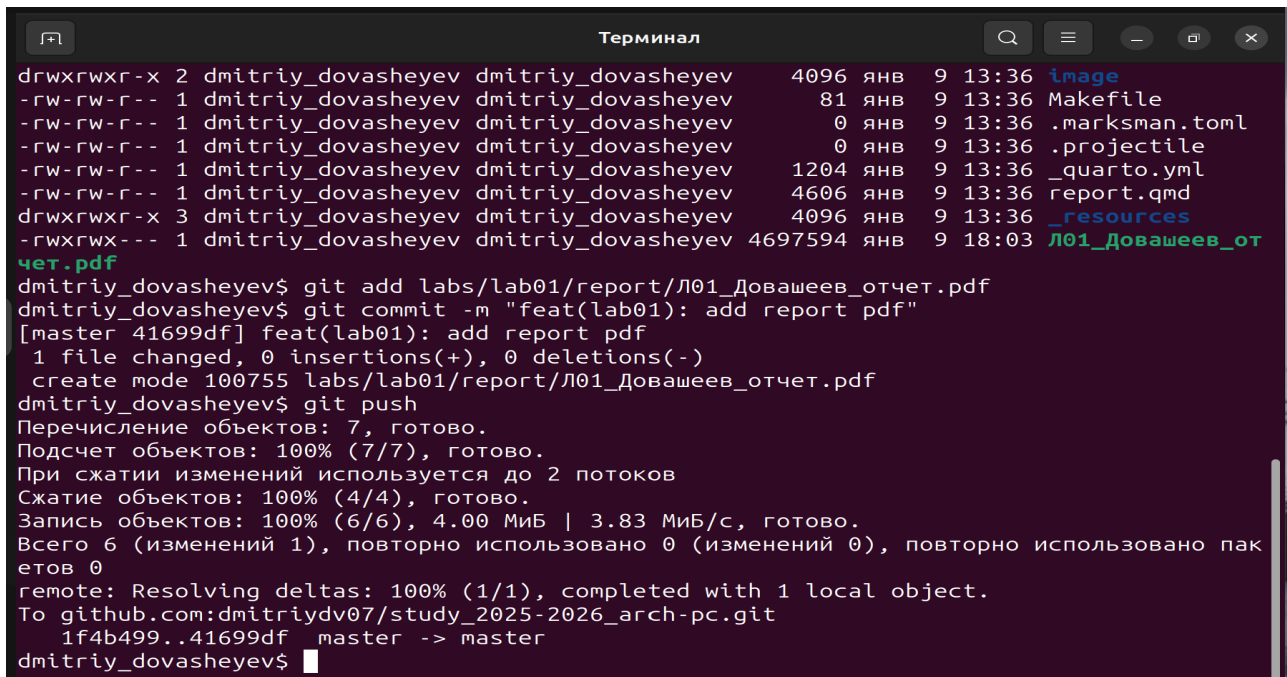
A terminal window titled "Терминал" with a dark background. The user, dmitriy_dovasheyev, adds the current directory to the git index with git add . and commits the changes with git commit -am "feat(main): make course structure". The commit message is displayed. Then, they push the changes to the remote repository with git push. The output shows the progress of the push, including object counting, compression, and the final push to the remote repository.

```
dmitriy_dovasheyev$ git add .
dmitriy_dovasheyev$ git commit -am "feat(main): make course structure"
[master 1f4b499] feat(main): make course structure
 2 files changed, 1 insertion(+), 14 deletions(-)
 delete mode 100644 package.json
dmitriy_dovasheyev$ git push
Перечисление объектов: 5, готово.
Подсчет объектов: 100% (5/5), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (2/2), готово.
Запись объектов: 100% (3/3), 295 байтов | 295.00 КиБ/с, готово.
Всего 3 (изменений 1), повторно использовано 0 (изменений 0), повторно использовано пак
етов 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:dmitriydv07/study_2025-2026_arch-pc.git
 b449d8f..1f4b499 master -> master
dmitriy_dovasheyev$
```

Рис 4.3.2 Отправка изменений на удаленный репозиторий.

5 Задания для самостоятельной работы.

Через терминал отправляю предыдущий отчет по лабораторной работе на свой удаленный репозиторий в GitHub (рис. 5.1), затем проверяю изменения на самом GitHub. (рис 5.2)



```
Терминал
drwxrwxr-x 2 dmitriy_dovasheyev dmitriy_dovasheyev 4096 янв 9 13:36 image
-rw-rw-r-- 1 dmitriy_dovasheyev dmitriy_dovasheyev 81 янв 9 13:36 Makefile
-rw-rw-r-- 1 dmitriy_dovasheyev dmitriy_dovasheyev 0 янв 9 13:36 .marksman.toml
-rw-rw-r-- 1 dmitriy_dovasheyev dmitriy_dovasheyev 0 янв 9 13:36 .projectile
-rw-rw-r-- 1 dmitriy_dovasheyev dmitriy_dovasheyev 1204 янв 9 13:36 _quarto.yml
-rw-rw-r-- 1 dmitriy_dovasheyev dmitriy_dovasheyev 4606 янв 9 13:36 report.qmd
drwxrwxr-x 3 dmitriy_dovasheyev dmitriy_dovasheyev 4096 янв 9 13:36 _resources
-rwxrwx--- 1 dmitriy_dovasheyev dmitriy_dovasheyev 4697594 янв 9 18:03 Л01_Довашеев_отчет.pdf
dmitriy_dovasheyev$ git add labs/lab01/report/Л01_Довашеев_отчет.pdf
dmitriy_dovasheyev$ git commit -m "feat(lab01): add report pdf"
[master 41699df] feat(lab01): add report pdf
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100755 labs/lab01/report/Л01_Довашеев_отчет.pdf
dmitriy_dovasheyev$ git push
Перечисление объектов: 7, готово.
Подсчет объектов: 100% (7/7), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (4/4), готово.
Запись объектов: 100% (6/6), 4.00 МиБ | 3.83 МиБ/с, готово.
Всего 6 (изменений 1), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:dmitriydv07/study_2025-2026_arch-pc.git
1f4b499..41699df master -> master
dmitriy_dovasheyev$
```

Рис 5.1 Отправка изменений на удаленный репозиторий.

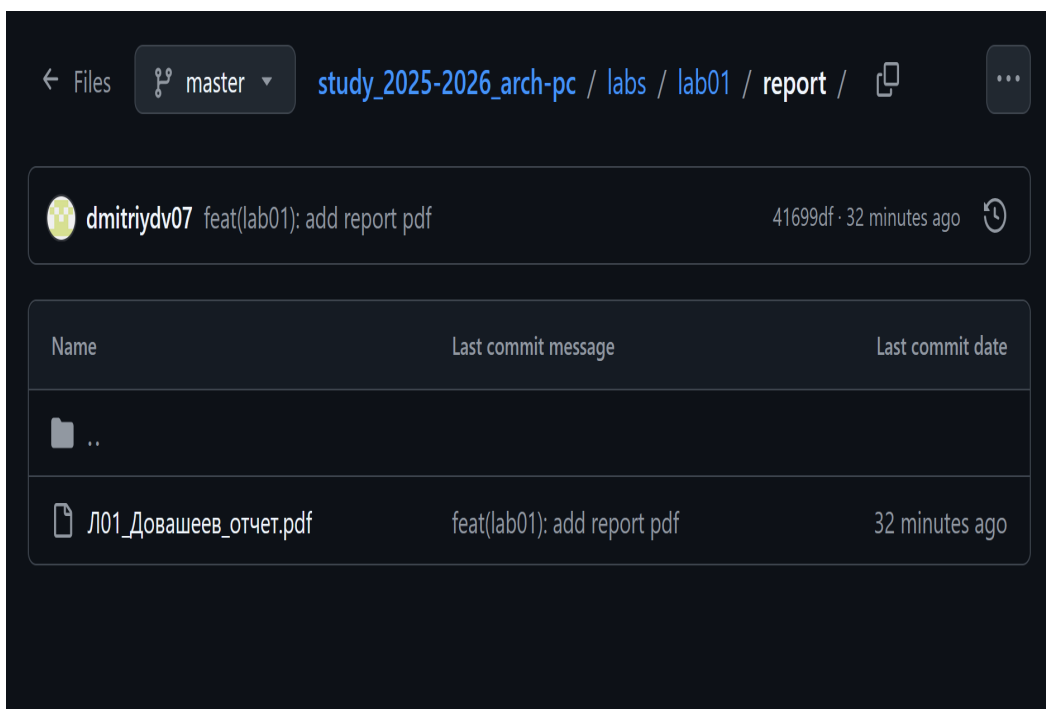


Рис 5.2 Проверка выполненных изменений на GitHub.

6 Выводы

При выполнении данной лабораторной работы я изучил идеологию и применение средств контроля версий, а также приобрел практические навыки по работе с реализацией VSC git.

Список литературы

1. https://esystem.rudn.ru/pluginfile.php/2089082/mod_resource/content/0/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%962.%20%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0%20%D0%BA%D0%BE%D0%BD%D1%82%D1%80%D0%BE%D0%BB%D1%8F%20%D0%B2%D0%B5%D1%80%D1%81%D0%B8%D0%B9%20Git.pdf
2. <https://esystem.rudn.ru/mod/page/view.php?id=1030492>
3. <https://esystem.rudn.ru/mod/resource/view.php?id=1030495>
4. <https://esystem.rudn.ru/mod/resource/view.php?id=1030496>
5. https://github.com/evdvorkina/study_2022-2023_arh-pc/tree/master