

# **Отчёт по лабораторной работе №2**

**Дисциплина: архитектура компьютера**

Цоппа Ева Эдуардовна

# Содержание

1 Цель работы.....	5
2 Задание.....	6
3 Теоретическое введение.....	7
4 Выполнение лабораторной работы .....	9
4.1 Настройка GitHub.....	9
4.2 Базовая настройка Git .....	9
4.3 Создание SSH-ключа .....	10
4.4 Создание рабочего пространства и репозитория курса на основе шаблона.....	13
4.5 Создание репозитория курса на основе шаблона .....	13
4.6 Настройка каталога курса.....	16
4.7 Выполнение заданий для самостоятельной работы .....	18
5 Выводы.....	22
6 Список литературы .....	23

# Список иллюстраций

4.1 Заполнение данных учетной записи GitHub .....	9
4.2 Аккаунт GitHub .....	9
4.3 Предварительная конфигурация git .....	10
4.4 Настройка кодировки .....	10
4.5 Создание имени для начальной ветки .....	10
4.6 Параметр autocrlf .....	10
4.7 Параметр safecrlf .....	10
4.8 Генерация SSH-ключа .....	11
4.9 Установка утилиты xclip.....	11
4.10 Копирование содержимого файла.....	12
4.11 Окно SSH and GPG keys .....	12
4.12 Добавление ключа .....	13
4.13 Создание рабочего пространства.....	13
4.14 Страница шаблона для репозитория .....	14
4.15 Окно создания репозитория .....	14
4.16 Созданный репозиторий .....	15
4.17 Перемещение между директориями .....	15
4.18 Клонирование репозитория .....	15
4.19 Окно с ссылкой для копирования репозитория .....	16
4.20 Перемещение между директориями .....	16
4.21 Удаление файлов.....	16
4.22 Создание каталогов .....	16

4.23 Добавление и сохранение изменений на сервере .....	17
4.24 Выгрузка изменений на сервер .....	17
4.25 Страница репозитория.....	18
4.26 Создание файла .....	18
4.27 Меню приложений .....	18
4.28 Работа с отчетом в текстовом процессоре .....	19
4.29 Перемещение между директориями.....	19
4.30 Проверка местонахождения файлов.....	19
4.31 Копирование файла .....	19
4.32 Добавление файла на сервер .....	20
4.33 Подкаталоги и файлы в репозитории .....	20
4.34 Отправка в центральный репозиторий сохраненных изменений.....	20
4.35 Страница каталога в репозитории .....	20
4.36 Страница последних изменений в репозитории.....	21
4.37 Каталог lab01/report.....	21
4.38 Каталог lab02/report .....	21

# 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.

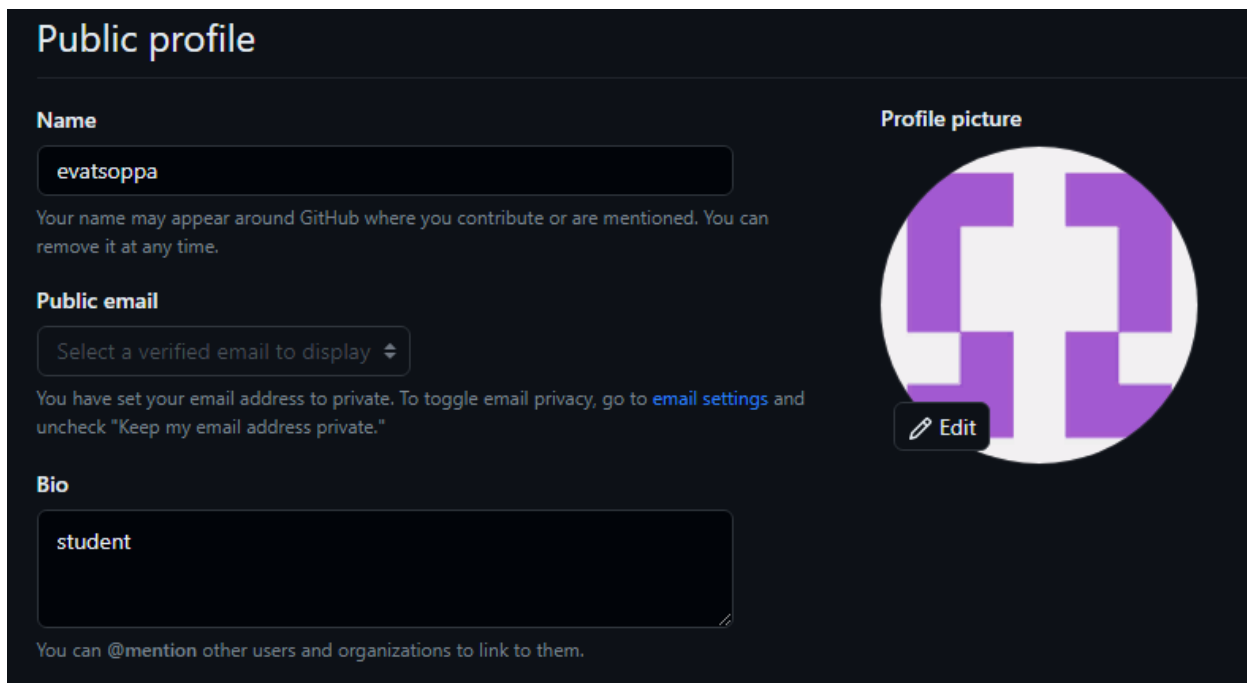
Обычно такого рода информация хранится в журнале изменений, доступ к которому можно ограничить. В отличие от классических, в распределённых системах контроля версий центральный репозиторий не является обязательным. Среди классических VCS наиболее известны CVS, Subversion, а среди распределённых — Git, Bazaar, Mercurial. Принципы их работы схожи, отличаются они в основном синтаксисом используемых в работе команд. 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. Работа пользователя со своей веткой начинается с проверки и получения изменений из центрального репозитория (при этом в локальное дерево до начала этой процедуры не должно было вноситься изменений). Затем можно вносить изменения в локальном дереве и/или ветке. После завершения внесения какого-то изменения в файлы и/или каталоги проекта необходимо разместить их в центральном репозитории.



## 4 Выполнение лабораторной работы

### 4.1 Настройка GitHub

Создаю учетную запись на сайте GitHub (рис. 4.1). Далее я заполнила основные данные учетной записи.



The screenshot shows the 'Public profile' setup page on GitHub. It includes fields for 'Name' (filled with 'evatsoppa'), 'Public email' (with a dropdown to 'Select a verified email to display'), and 'Bio' (filled with 'student'). A 'Profile picture' section shows a placeholder image with an 'Edit' button. A note at the bottom states: 'You can @mention other users and organizations to link to them.'

Рис. 4.1. Заполнение данных учетной записи GitHub

Аккаунт создан (рис. 4.2).

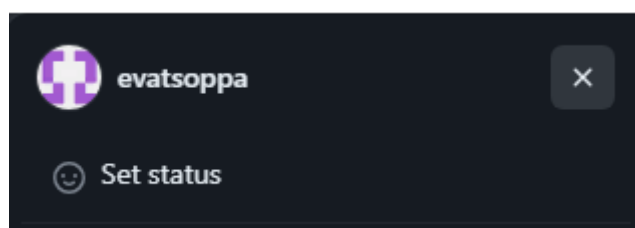


Рис. 4.2 Аккаунт GitHub

### 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, то есть мою (рис. 4.3).

```
[evatsoppa@evatsoppa ~]$ git config --global user.name "<Eva Tsoppa>"
[evatsoppa@evatsoppa ~]$ git config --global user.email "<1132236045@pfur.ru>"
```

Рис. 4.3. Предварительная конфигурация git

Настраиваю utf-8 в выводе сообщений git для корректного отображения символов (рис. 4.4).

```
[evatsoppa@evatsoppa ~]$ git config --global core.quotepath false
```

Рис. 4.4. Настройка кодировки

Задаю имя «master» для начальной ветки (рис. 4.5).

```
[evatsoppa@evatsoppa ~]$ git config --global init.defaultBranch master
```

Рис. 4.5. Создание имени для начальной ветки

Задаю параметр autocrlf со значением input, так как я работаю в системе Linux, чтобы конвертировать CRLF в LF только при коммитах (рис. 4.6). CR и LF – это символы, которые можно использовать для обозначения разрыва строки в текстовых файлах.

```
[evatsoppa@evatsoppa ~]$ git config --global core.autocrlf input
```

Рис. 4.6 Параметр autocrlf

Задаю параметр safecrlf со значением warn, так Git будет проверять преобразование на обратимость (рис. 4.7). При значении warn Git только выведет предупреждение, но будет принимать необратимые конвертации.

```
[evatsoppa@evatsoppa ~]$ git config --global core.safecrlf warn
```

Рис. 4.7. Параметр safecrlf

## 4.3 Создание SSH-ключа

Для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый). Для этого ввожу команду `ssh-keygen -C "Имя Фамилия, work@email"`, указывая имя владельца и

электронную почту владельца (рис. 4.8). Ключ автоматически сохранится в каталоге ~/.ssh/.

```
[evatsoppa@evatsoppa ~]$ ssh-keygen -C "Eva Tsoppa <1132236045@pfur.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/evatsoppa/.ssh/id_rsa):
Created directory '/home/evatsoppa/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/evatsoppa/.ssh/id_rsa
Your public key has been saved in /home/evatsoppa/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:PtXB70JF74LdgrDJu/9/yXCy4l2MC3AT7WhUtu4S6Vc Eva Tsoppa <1132236045@pfur.ru>
The key's randomart image is:
+----[RSA 3072]-----+
|
|      o
|    o+ .
|   o=o
|  .+*o
| S.oB+o.E
| o *=*+=+.
| * ++++0o.
| o .+=+oo.
| ooooo+.
+-----[SHA256]-----+
[evatsoppa@evatsoppa ~]$
```

Рис. 4.8. Генерация SSH-ключа

Xclip – утилита, позволяющая скопировать любой текст через терминал. Сначала провожу ее установку. (рис. 4.9).

```
Установить пакет «xclip», предоставляющий команду «xclip»? [N/y] y

* Ожидание в очереди...

* Загрузка списка пакетов....
Следующие пакеты должны быть установлены:
xclip-0.13-19.git11cba61.fc38.x86_64 Command line clipboard grabber
Продолжить с этими изменениями? [N/y] y

* Ожидание в очереди...
* Ожидание аутентификации...
* Ожидание в очереди...
* Загрузка пакетов...
* Запрос данных...
* Проверка изменений...
* Установка пакетов...
```

Рис. 4.9. Установка утилиты xclip

Копирую открытый ключ из директории, в которой он был сохранен, с помощью утилиты xclip (рис. 4.10).

```
[evatsoppa@evatsoppa ~]$ cat ~/.ssh/id_rsa.pub | xclip -sel clip
```

Рис. 4.10. 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. 4.11).

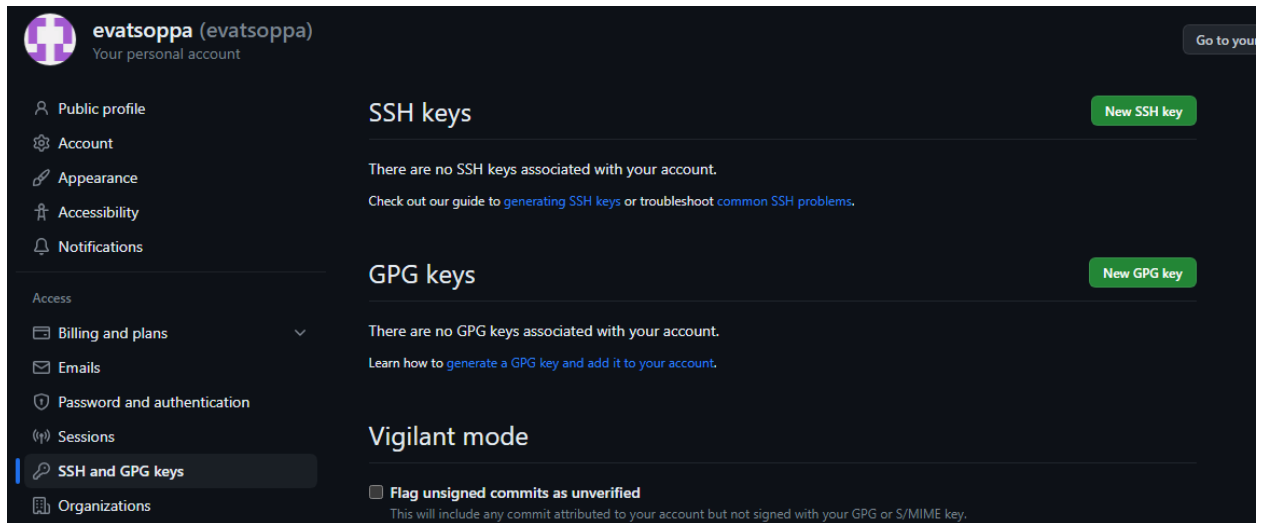


Рис. 4.11. Окно SSH and GPG keys

Вставляю скопированный ключ в поле «Key». В поле Title указываю имя для ключа. Нажимаю «Add SSH-key», чтобы завершить добавление ключа (рис. 4.12).

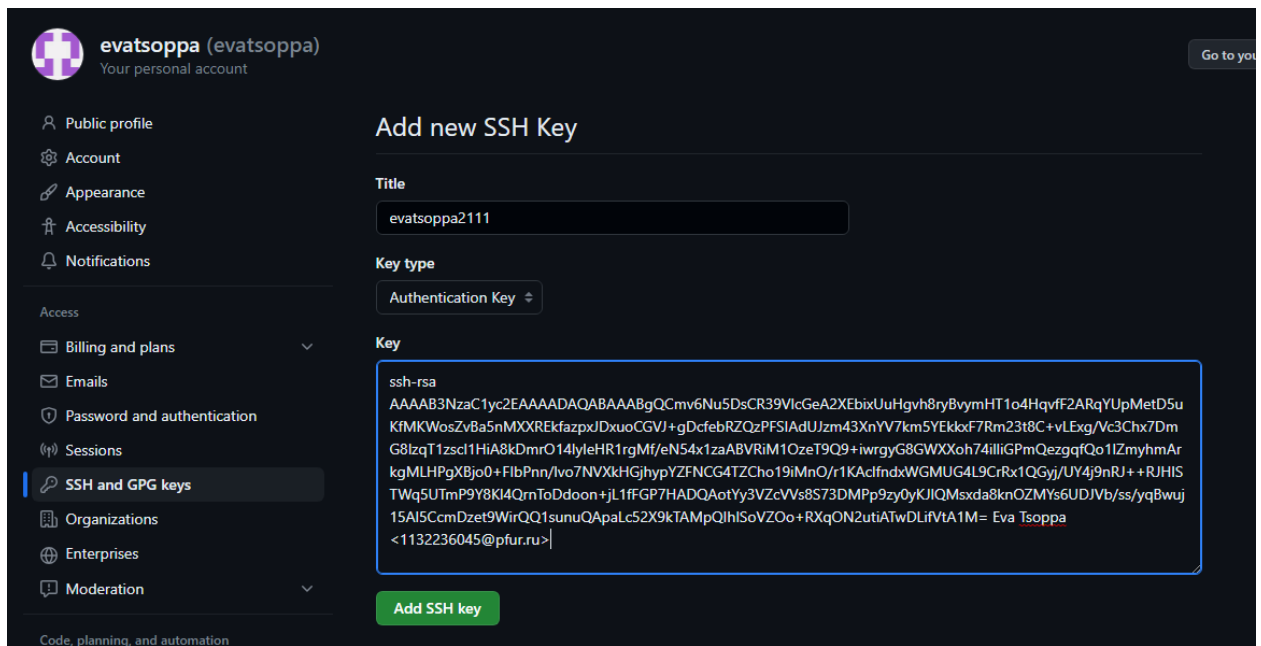


Рис. 4.12. Добавление ключа

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

Закрываю браузер, открываю терминал. Создаю директорию, рабочее пространство, с помощью утилиты `mkdir`, благодаря ключу `-p` создаю все директории после домашней `~/work/study/2023-2024/“Архитектура компьютера”` рекурсивно. Далее проверяю с помощью `ls`, действительно ли были созданы необходимые мне каталоги (рис. 4.13).

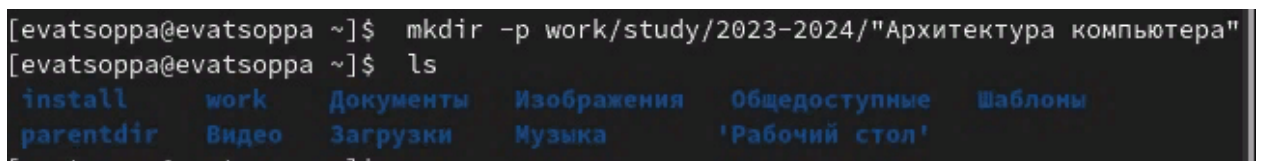


Рис. 4.13. Создание рабочего пространства

## 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 (рис. 4.14).

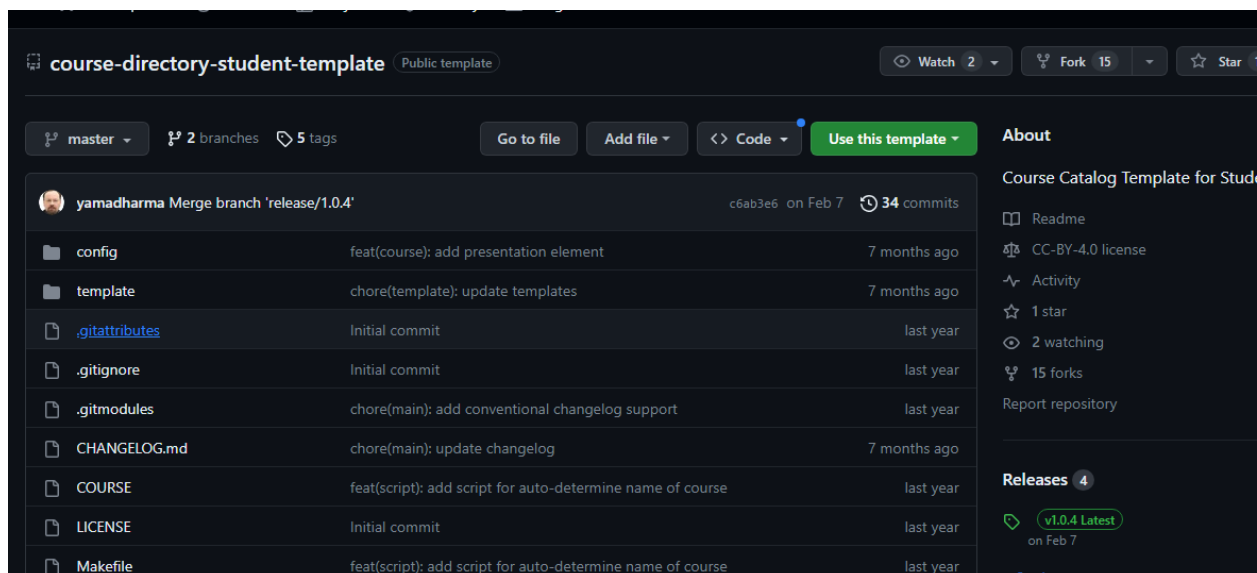


Рис. 4.14. Страница шаблона для репозитория

В открывшемся окне задаю имя репозитория (Repository name): study\_2023–2024\_arhpc и создаю репозиторий, нажимаю на кнопку «Create repository from template» (рис. 4.15).

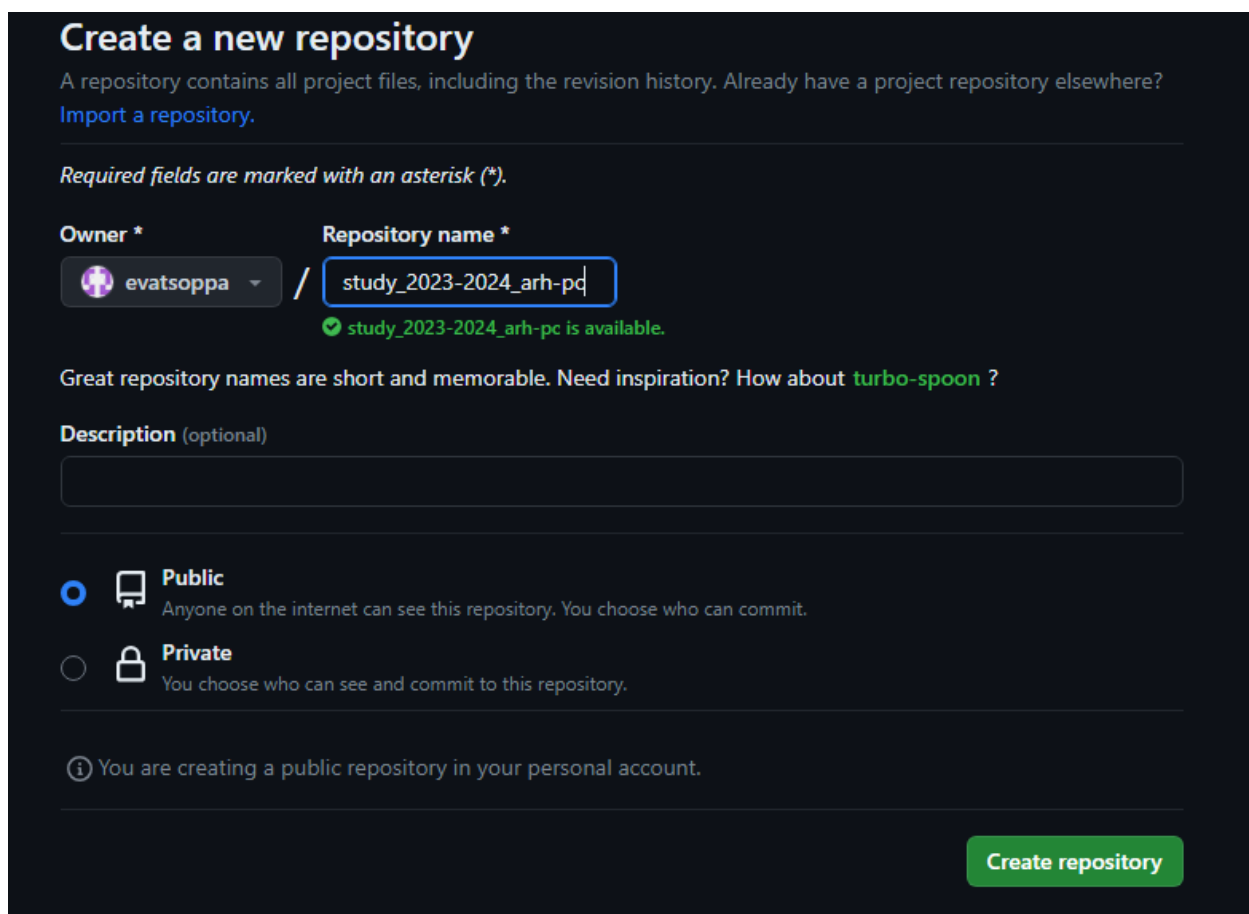


Рис. 4.15. Окно создания репозитория

Репозиторий создан (рис. 4.16).

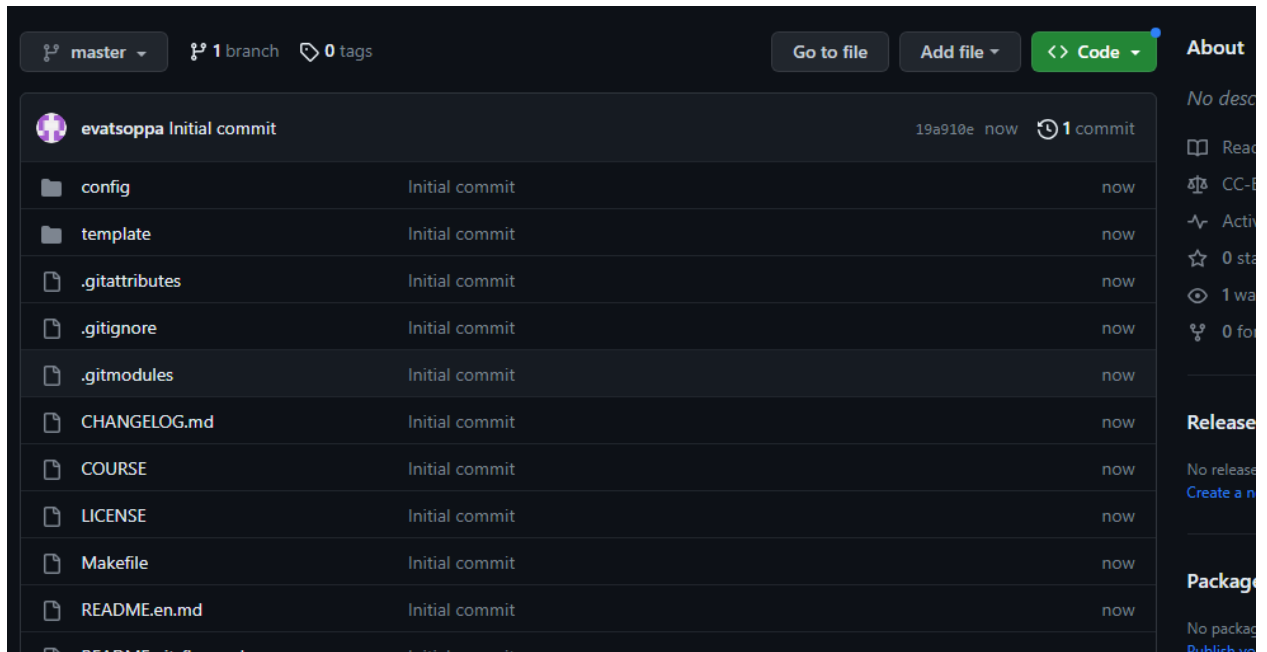


Рис. 4.16. Созданный репозиторий

Через терминал перехожу в созданный каталог курса с помощью утилиты `cd` (рис. 4.17).

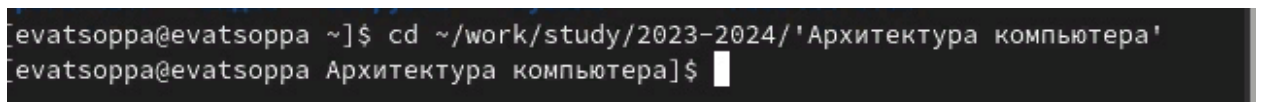


Рис. 4.17. Перемещение между директориями

Клонирую созданный репозиторий с помощью команды `git clone --recursive git@github.com:study_2023-2024_arh-pc.git arch-pc` (рис. 4.18).

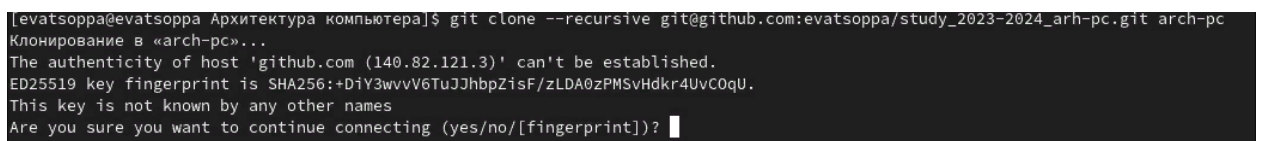


Рис. 4.18. Клонирование репозитория

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH» (рис. 4.19).

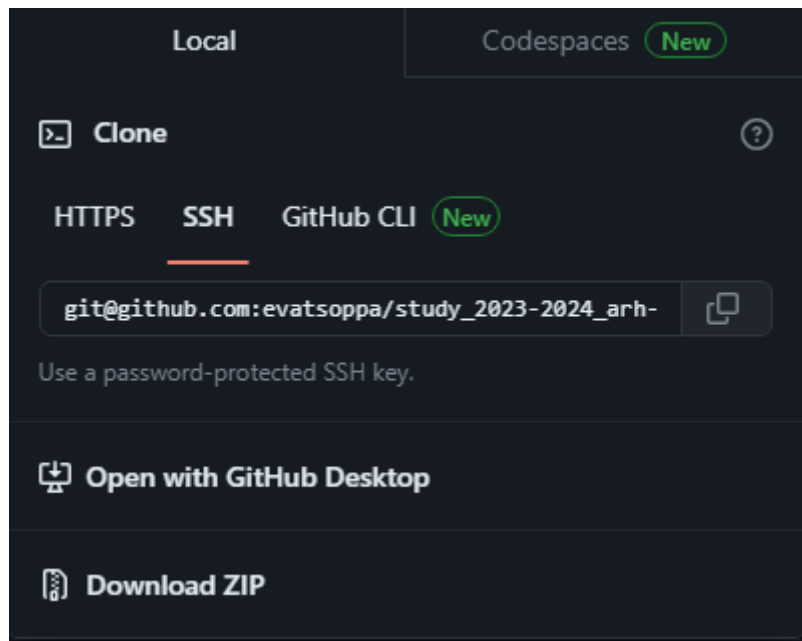


Рис. 4.19. Окно с ссылкой для копирования репозитория

## 4.6 Настройка каталога курса

Перехожу в каталог arch-pc с помощью утилиты cd (рис. 4.20).

```
[evatsoppa@evatsoppa Архитектура компьютера]$ cd ~/work/study/2023-2024/Архитектура\ компьютера/arch-pc  
[evatsoppa@evatsoppa arch-pc]$
```

Рис. 4.20. Перемещение между директориями

Удаляю лишние файлы с помощью утилиты rm (рис. 4.21).

```
[evatsoppa@evatsoppa arch-pc]$ rm package.json
```

Рис. 4.21. Удаление файлов

Создаю необходимые каталоги (рис. 4.22).

```
[evatsoppa@evatsoppa arch-pc]$ echo arch-pc > COURSE  
[evatsoppa@evatsoppa arch-pc]$ make
```

Рис. 4.22. 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. 4.23).



```

[evatsoppa@evatsoppa arch-pc]$ git add .
[evatsoppa@evatsoppa arch-pc]$ git commit -am 'feat(main): make course structure'
[master 94b6efd] feat(main): make course structure
199 files changed, 54725 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/image/kulyabov.jpg
create mode 100644 labs/lab01/presentation/presentation.md
create mode 100644 labs/lab01/report/Makefile
create mode 100644 labs/lab01/report/bib/cite.bib
create mode 100644 labs/lab01/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab01/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_fignos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_secnos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/core.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/main.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 labs/lab01/report/report.md
create mode 100644 labs/lab02/presentation/Makefile
create mode 100644 labs/lab02/presentation/image/kulyabov.jpg
create mode 100644 labs/lab02/presentation/presentation.md
create mode 100644 labs/lab02/report/Makefile
create mode 100644 labs/lab02/report/bib/cite.bib
create mode 100644 labs/lab02/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab02/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_fignos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_secnos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/__init__.py

```

Рис. 4.23. Добавление и сохранение изменений на сервере

Отправляю все на сервер с помощью push (рис. 4.24).

```

[evatsoppa@evatsoppa arch-pc]$ git push
Перечисление объектов: 37, готово.
Подсчет объектов: 100% (37/37), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (29/29), готово.
Запись объектов: 100% (35/35), 342.13 КиБ | 1.31 МиБ/с, готово.
Всего 35 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:evatsoppa/study_2023-2024_arh-pc.git
 19a910e..94b6efd master -> master

```

Рис. 4.24. Выгрузка изменений на сервер

Проверяю правильность выполнения работы сначала на самом сайте GitHub (рис. 4.25).

Eva Tsoppa feat(main): make course structure		94b6efd · 4 minutes ago	History
Name	Last commit message	Last commit date	
..			
lab01	feat(main): make course structure	4 minutes ago	
lab02	feat(main): make course structure	4 minutes ago	
lab03	feat(main): make course structure	4 minutes ago	
lab04	feat(main): make course structure	4 minutes ago	
lab05	feat(main): make course structure	4 minutes ago	
lab06	feat(main): make course structure	4 minutes ago	
lab07	feat(main): make course structure	4 minutes ago	
lab08	feat(main): make course structure	4 minutes ago	
lab09	feat(main): make course structure	4 minutes ago	
lab10	feat(main): make course structure	4 minutes ago	
lab11	feat(main): make course structure	4 minutes ago	

Рис. 4.25. Страница репозитория

## 4.7 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. 4.26).

```
[evatsoppa@evatsoppa arch-pc]$ cd ~/work/study/2023-2024/Архитектура\ компьютера/arch-pc/labs/lab02/report
[evatsoppa@evatsoppa report]$ touch Л02_Цонна_отчет
```

Рис. 4.26. Создание файла

Оформить отчет я смогу в текстовом процессоре LibreOffice Writer, найдя его в меню приложений (рис. 4.27).

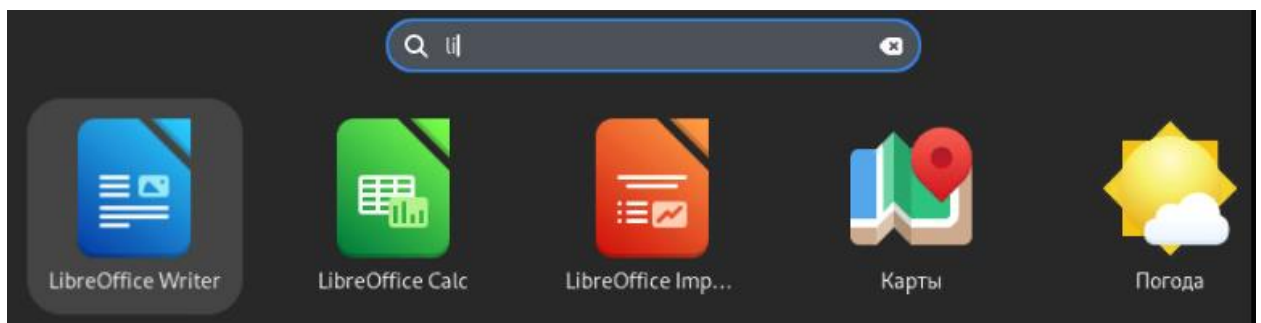


Рис. 4.27. Меню приложений

После открытия текстового процессора открываю в нем созданный файл и могу начать в нем работу над отчетом (рис. 4.28).

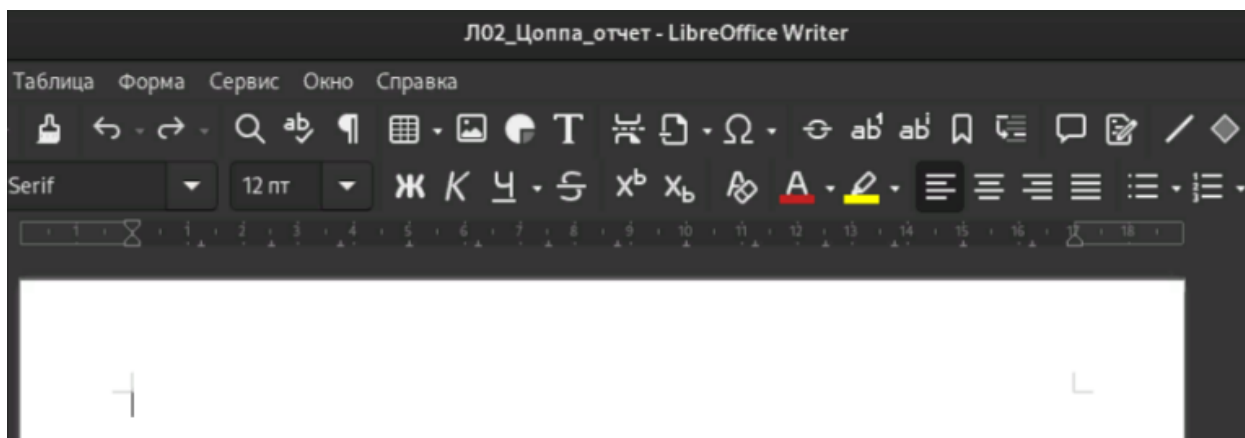


Рис. 4.28. Работа с отчетом в текстовом процессоре

2. Перехожу из подкаталога lab02/report в подкаталог lab01/report с помощью утилиты cd (рис. 4.29).

```
[evatsoppa@evatsoppa report]$ cd ..
[evatsoppa@evatsoppa lab02]$ cd ..
[evatsoppa@evatsoppa labs]$ cd lab01/
[evatsoppa@evatsoppa lab01]$ cd ..
[evatsoppa@evatsoppa labs]$ cd lab01/report/
[evatsoppa@evatsoppa report]$
```

Рис.4.29. Перемещение между директориями

Проверяю местонахождение файлов с отчетами по первой и второй лабораторным работам. Они должны быть в подкаталоге домашней директории «Загрузки», для проверки использую команду ls (рис. 4.30).

```
[evatsoppa@evatsoppa report]$ cp ~/Загрузки/Л01_Цоппа_отчет.pdf /home/evatsoppa/work/study/2023-2024/'Архитектура компьютера'/arch-pc/labs/lab01/report
[evatsoppa@evatsoppa report]$ ls
* bib image Makefile pandoc report.md Л01_Цоппа_отчет.pdf
```

Рис. 4.30. Проверка местонахождения файлов

Копирую первую лабораторную с помощью утилиты cp и проверяю правильность выполнения команды cp с помощью ls (рис. 4.31).

```
[evatsoppa@evatsoppa report]$ cp ~/Загрузки/Л01_Цоппа_отчет.pdf /home/evatsoppa/work/study/2023-2024/'Архитектура компьютера'/arch-pc/labs/lab01/report
[evatsoppa@evatsoppa report]$ ls
* bib image Makefile pandoc report.md Л01_Цоппа_отчет.pdf
```

Рис. 4.31. Копирование файлов

3. Добавляю с помощью команды git add в коммит созданные файлы: Л01\_Цоппа\_отчет (рис. 4.32).

```
[evatsoppa@evatsoppa report]$ git add /01_Цонна_отчет.pdf
```

Рис.4.32. Добавление файла на сервер

Сохраняю изменения на сервере командой `git commit -m "..."`, поясняя, что добавила файлы.

То же самое делаю для отчета по второй лабораторной работе: перехожу в директорию `labs/lab02/report` с помощью `cd`, добавляю с помощью `git add` нужный файл, сохраняю изменения с помощью `git commit` (рис. 4.33).

```
[evatsoppa@evatsoppa arch-pc]$ cd labs/lab02/report
[evatsoppa@evatsoppa report]$ git add /02_Цонна_отчет
[evatsoppa@evatsoppa report]$ git commit -m "Add existing file"
[master f8e0e74] Add existing file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 labs/lab02/report/02_Цонна_отчет
```

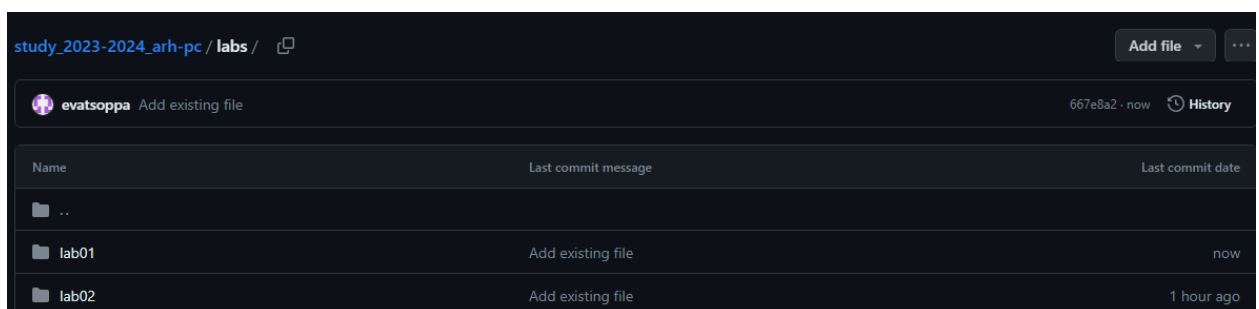
Рис.4.33. Подкаталоги и файлы в репозитории

Отправляю в центральный репозиторий сохраненные изменения командой `git push -f origin master` (рис. 4.34).

```
[evatsoppa@evatsoppa report]$ git push -f origin master
Перечисление объектов: 67, готово.
Подсчет объектов: 100% (67/67), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (59/59), готово.
Запись объектов: 100% (67/67), 359.39 КиБ | 1.16 МиБ/с, готово.
Всего 67 (изменений 8), повторно использовано 26 (изменений 1), повторно использовано пакетов 0
remote: Resolving deltas: 100% (8/8), done.
To github.com:evatsoppa/study_2023-2024_arh-pc.git
+ a09fb88...f8e0e74 master -> master (forced update)
```

Рис.4.34. Отправка в центральный репозиторий сохраненных изменений

Проверяю на сайте GitHub правильность выполнения заданий. Вижу, что пояснение к совершенным действиям отображается (рис. 4.35).



Name	Last commit message	Last commit date
..		
lab01	Add existing file	now
lab02	Add existing file	1 hour ago

Рис.4.35. Страница каталога в репозитории

При просмотре изменений так же вижу, что был добавлен файл с отчетом по лабораторной работе (рис. 4.36).

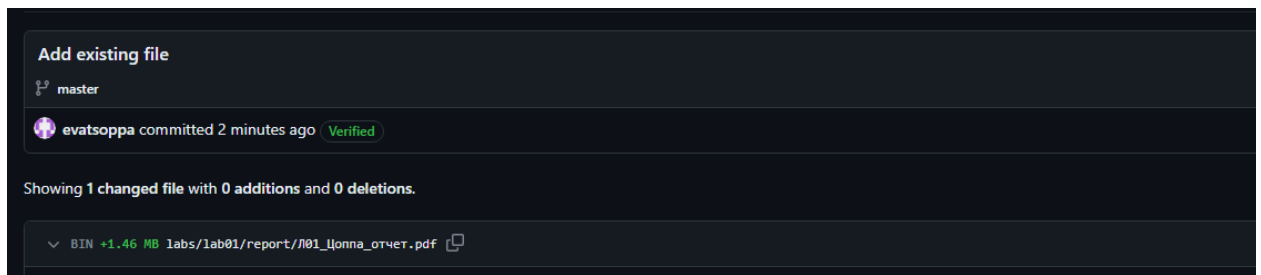


Рис.4.36. Страница последних изменений в репозитории

Вижу, что отчеты по лабораторным работам находятся в соответствующих каталогах репозитория: отчет по первой - в lab01/report (рис. 4.37), по второй – в lab02/report (рис. 4.38).

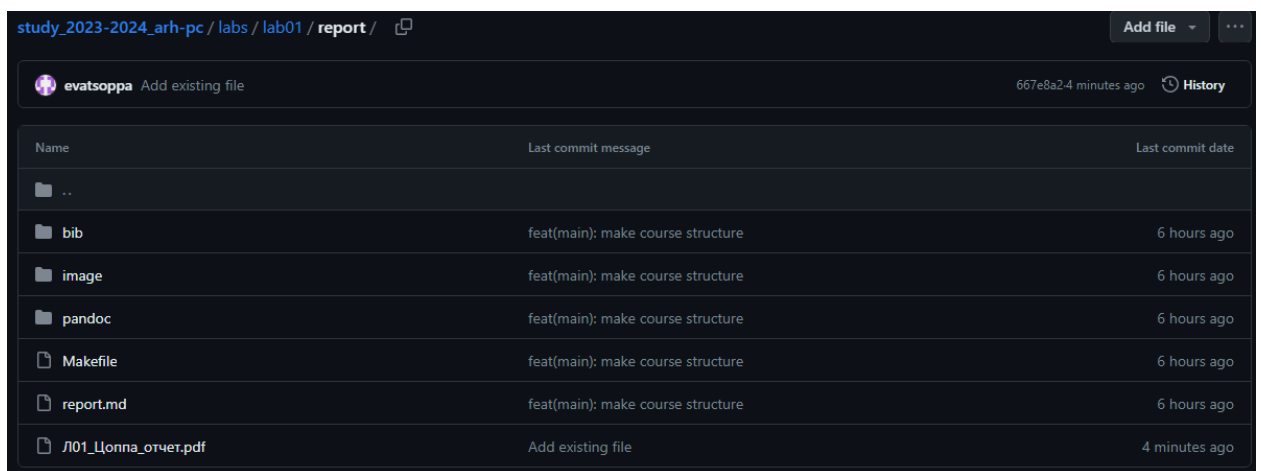


Рис.4.37. Каталог lab01/report

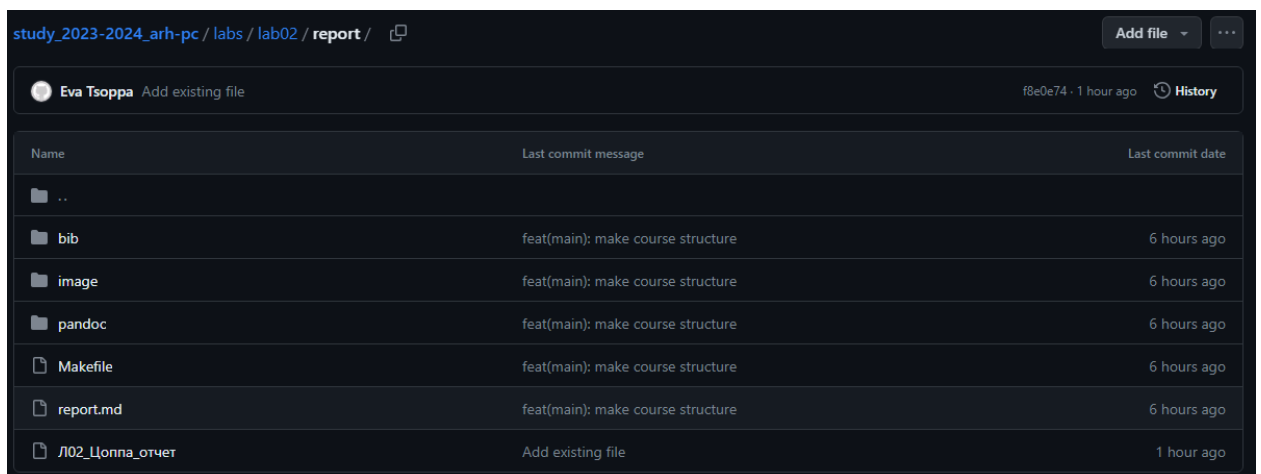


Рис.4.38. Каталог lab02/report

## **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.

## **6 Список литературы**

1. Архитектура ЭВМ
2. Git - gitattributes Документация