Отчёт по лабораторной работе №2

Дисциплина: Архитектура компьютера

Гасанова Шакира Чингизовна

Содержание

1 Цель работы .........................................................................................…………………………………………...3

2 Задание .................................................................................................……………………………………………..4

3 Теоретическое введение ...................................................................……………………………………...5

4 Выполнение лабораторной работы

4.1 Настройка GitHub................................................................................………………………………………..7

4.2 Базовая настойка Git...........................................................................……………………………………...7

4.3 Создание SSH-ключа...........................................................................……………………………………...8

4.4 Создание рабочего пространства и репозитория курса на

основе шаблона........................................................................................……………………………………….10

4.5 Создание репозитория курса на основе шаблона............................……………………10

4.6 Настройка каталога курса..............................................................………………………………......13

4.7 Выполнение заданий для самостоятельной работы....................………………........14

5 Выводы ................................................................................…………………………………………..............…...18

6 Источники ....................................................................................………………………………………….........19

# 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

5тому, 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. 1). Далее я заполнила основные

данные учетной записи. (рис. 1).

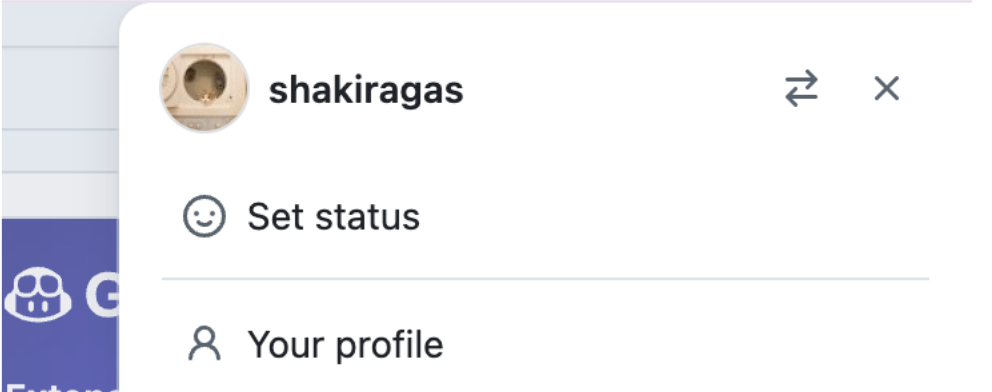


Рис. 1: Создание учетной записи

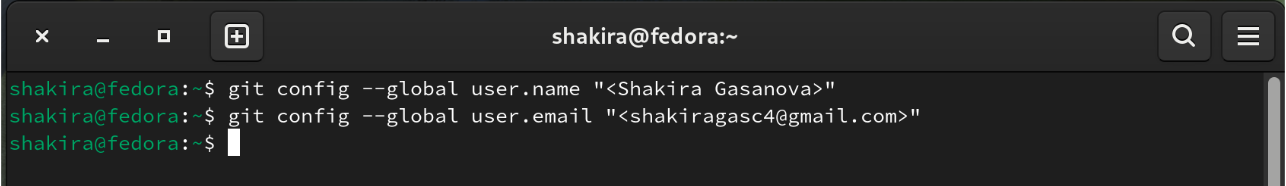
4.2 Базовая настройка Git

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. 2).

Рис. 2: Предварительная конфигурация гит

Настраиваю utf-8 в выводе сообщений git для корректного отображения

символов (рис. 3).

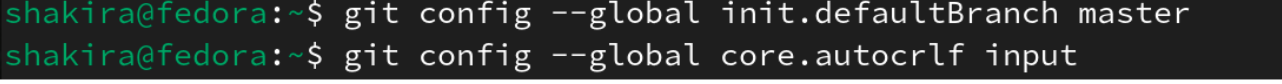
Рис. 3: Настройка кодировки

Задаю имя «master» для начальной ветки и задаю параметр autocrlf со

значением input, так как я работаю в системе Linux, чтобы конвертировать

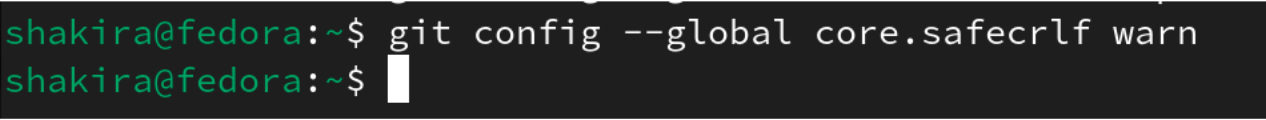
CRLF в LF только при коммитах. CR и LF – это символы, которые можно

использовать для обозначения разрыва строки в текстовых файлах (рис. 4).

Рис. 4: Создание имени для начальной ветки и задание параметра autocrlf

Задаю параметр safecrlf со значением warn, так Git будет проверять

преобразование на обратимость (рис. 5). При значении warn Git только выведет предупреждение, но будет принимать необратимые конвертации (рис. 5).

Рис. 5: Параметр safecrlf

4.3 Создание SSH-ключа

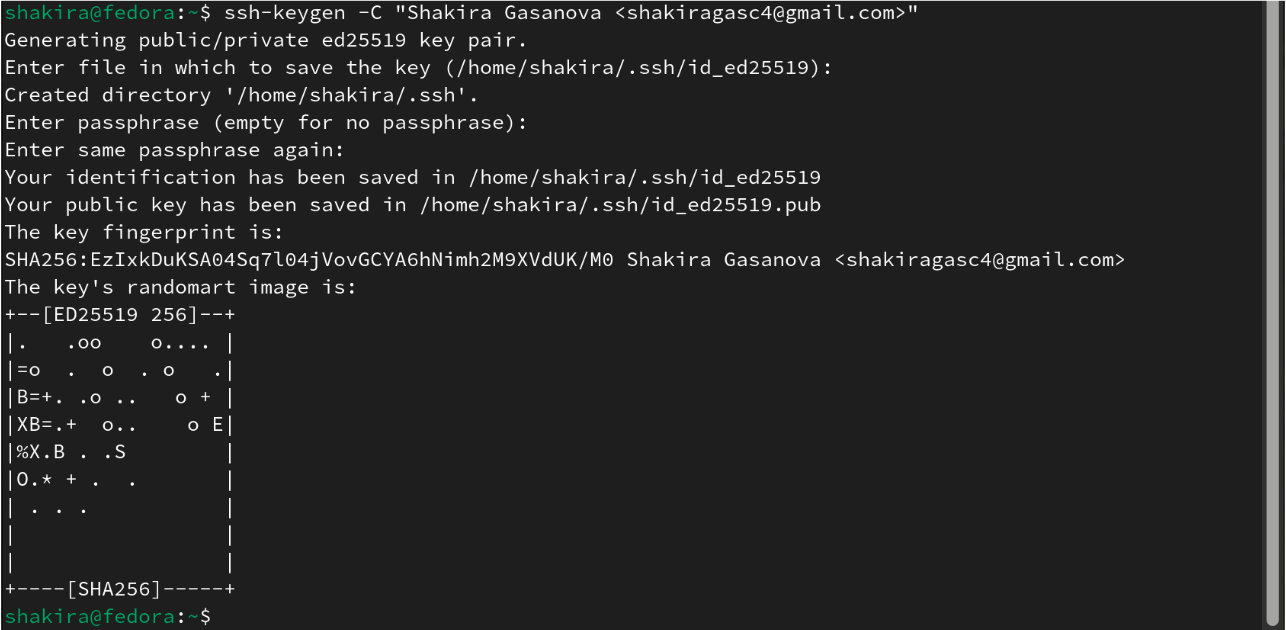
Для последующей идентификации пользователя на сервере репозиториев

требуется сгенерировать пару ключей (приватный и публичный). Для этого

использую команду ssh-keygen -C "Имя Фамилия, work@email", где

указываются имя владельца и его электронная почта. Сгенерированный ключ

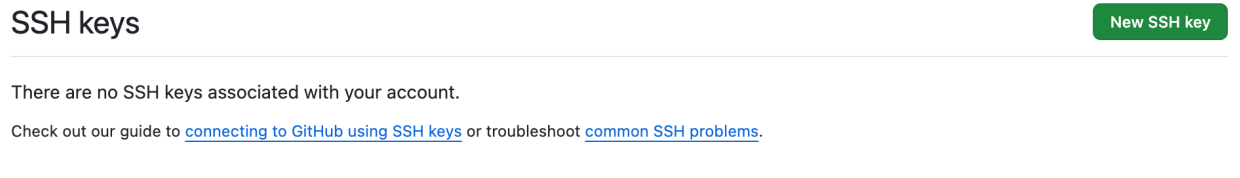
будет автоматически сохранён в каталоге ~/.ssh/ (рис. 6).

Рис. 6: Параметр safecrlf

Открываю браузер, захожу на сайт GitHub. Открываю свой профиль и

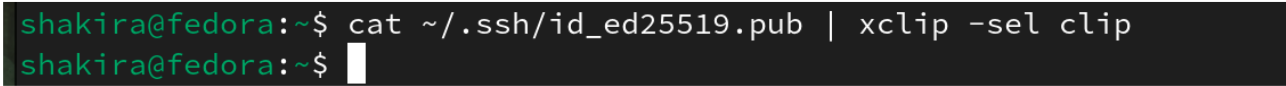
выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key»

(рис. 7).

Рис. 7: Окно SSH

Копирую открытый ключ из директории, в которой он был сохранен, с

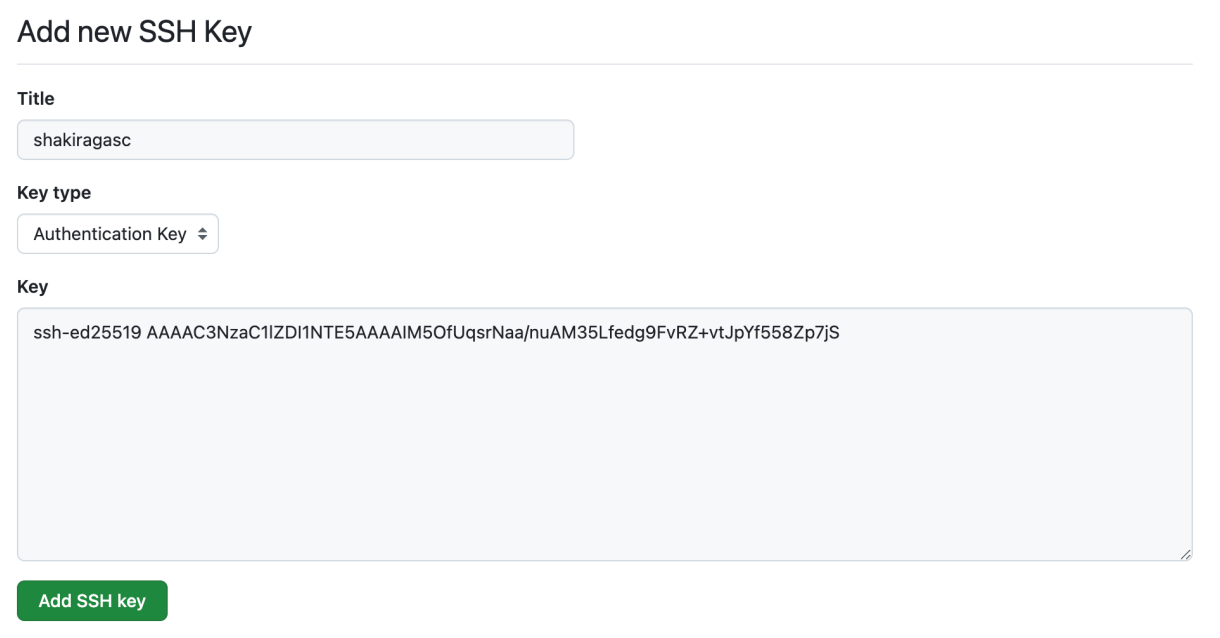
помощью утилиты xclip (рис. 8).

Рис. 8: Копирование ключа

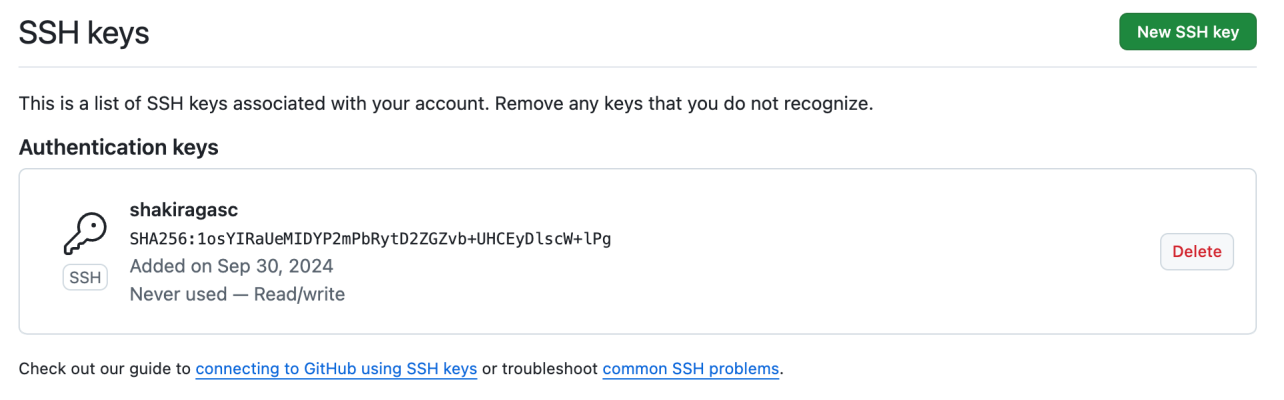
Вставляю скопированный ключ в поле «Key». В поле Title указываю имя

для ключа. Нажимаю «Add SSH-key», чтобы завершить добавление ключа

(рис. 9).

Рис. 9: Добавление ключа

Теперь появился ключ (рис. 10).

Рис. 10: Созданный SSH-ключ

4.4 Создание рабочего пространства и репозитория курса

на основе шаблона

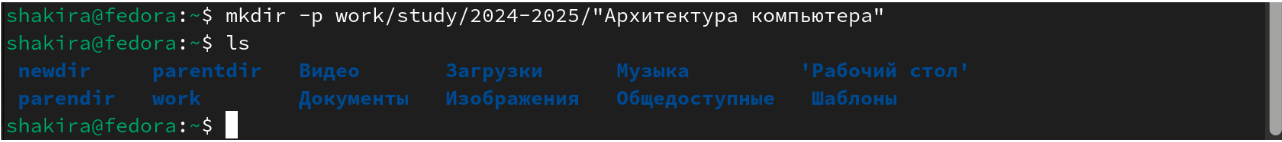
Закрываю браузер и открываю терминал. С помощью утилиты mkdir

создаю рабочую директорию. Использую ключ -p для рекурсивного создания

всех необходимых каталогов по пути ~/work/study/2024-2025/“Архитектура

компьютера”. Затем проверяю с помощью команды ls, были ли успешно

созданы нужные каталоги (рис. 11).

Рис. 11: Создание рабочего пространства

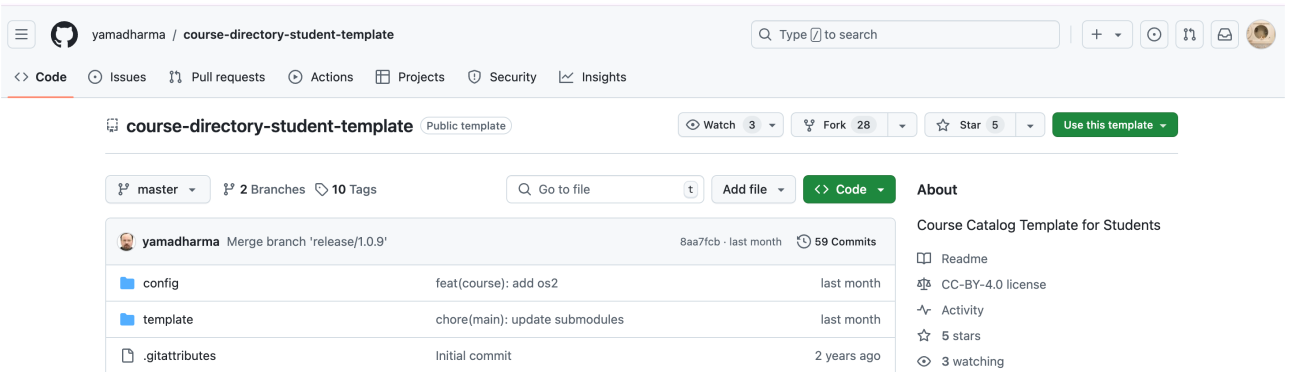
4.5 Создание репозитория курса на основе шаблона

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

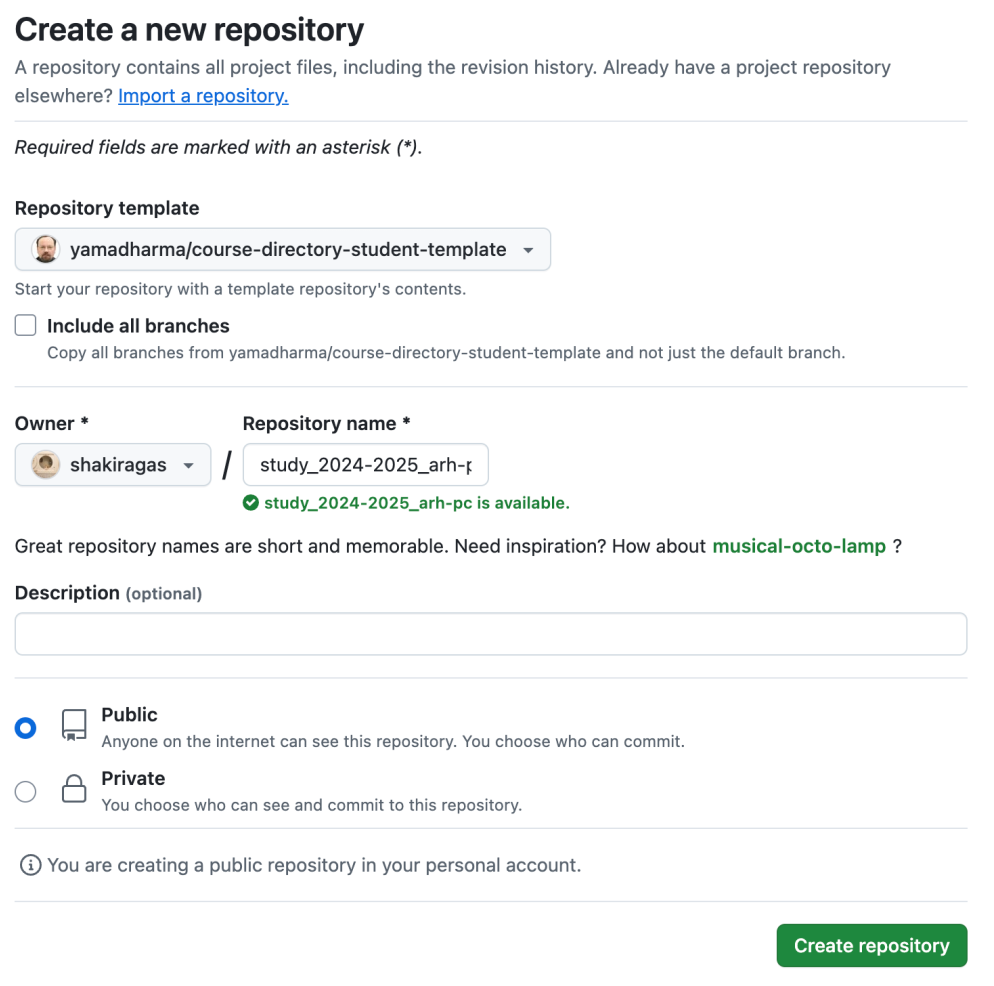
(рис. 12).

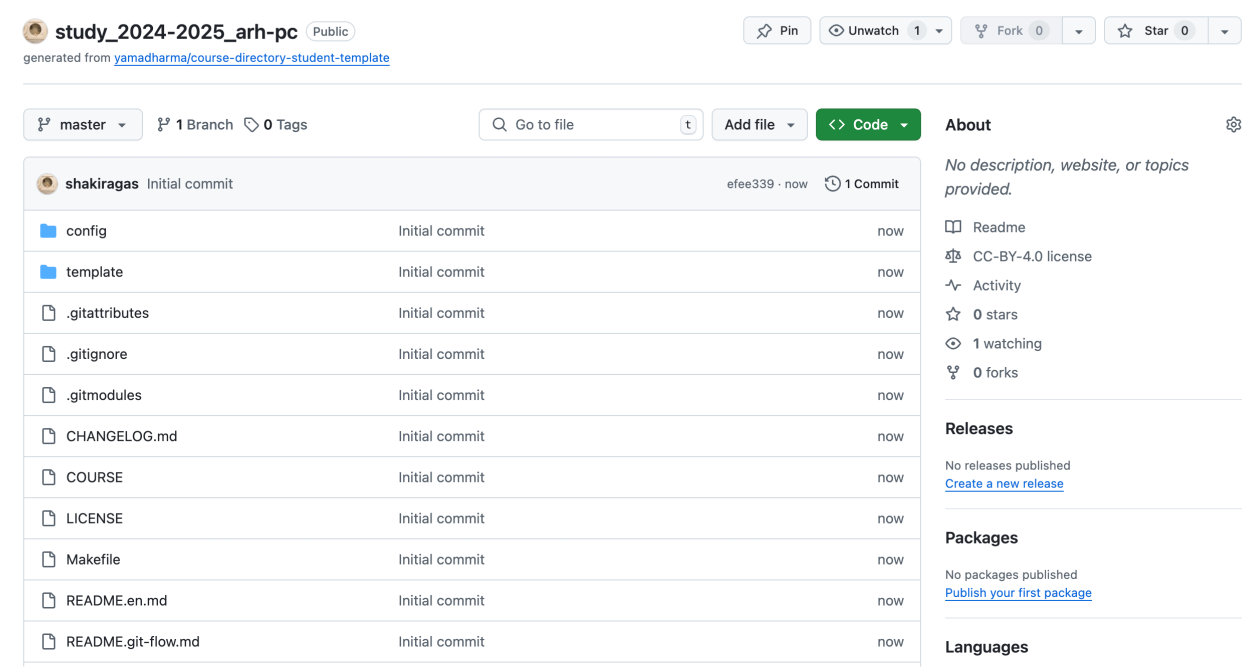
Рис. 12: Шаблон для репозитория

В открывшемся окне задаю имя репозитория (Repository name):

study\_2024–2025\_arh-pc и создаю репозиторий, нажимаю на кнопку «Create

repository from template» (рис. 13).

Рис. 13: Окно создания репозитория

Рис. 14: Созданный репозиторий

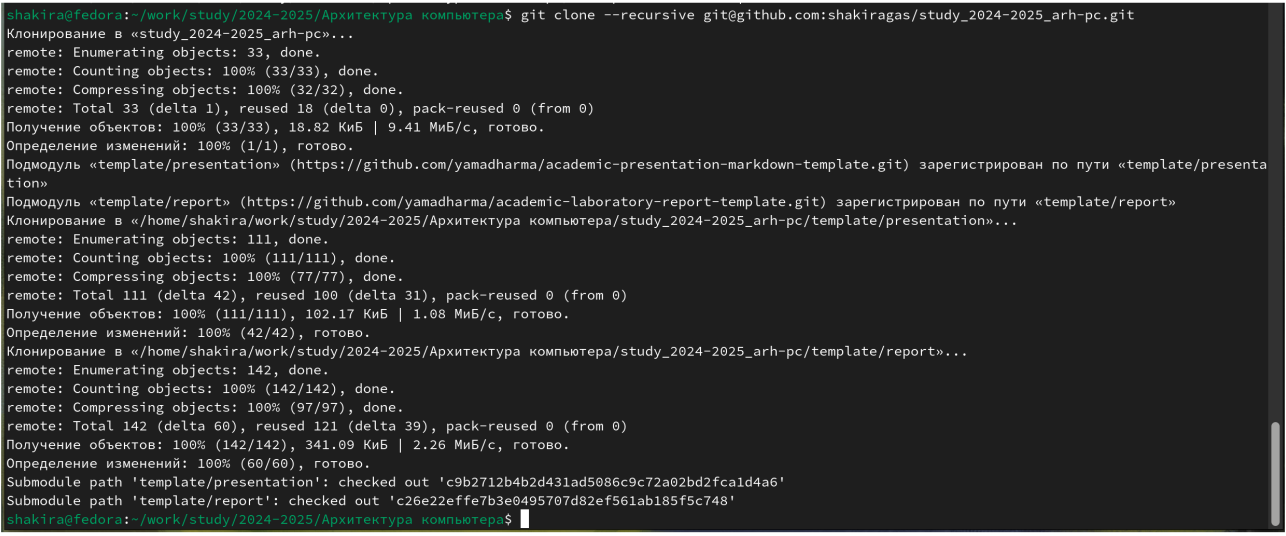
Через терминал перехожу в созданный каталог курса с помощью утилиты

cd (рис. 15).

Рис. 15: Переход в каталог курса

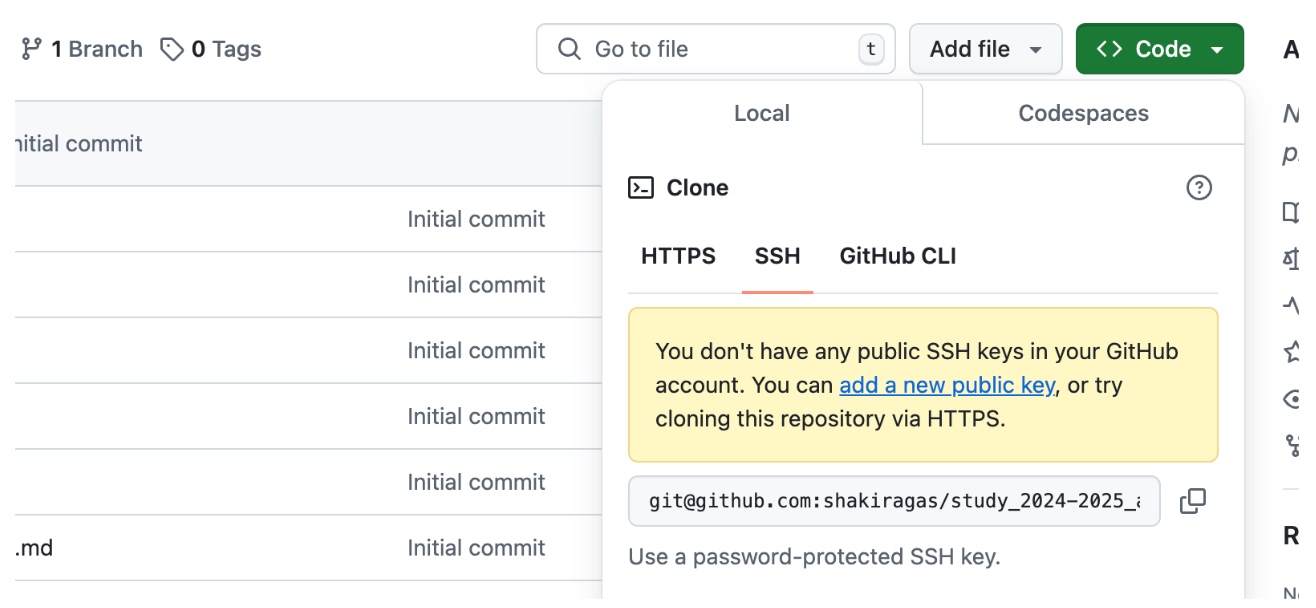
Клонирую созданный репозиторий с помощью команды git clone

-recursive git@github.com:/study\_2022–2023\_arh-pc.git arch-pc (рис. 16).

Рис. 16: Клонирование репозитория

Копирую ссылку для клонирования на странице созданного репозитория,

сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH» (рис. 17).

Рис. 17: Копирование репозитория

4.6 Настройка каталога курса

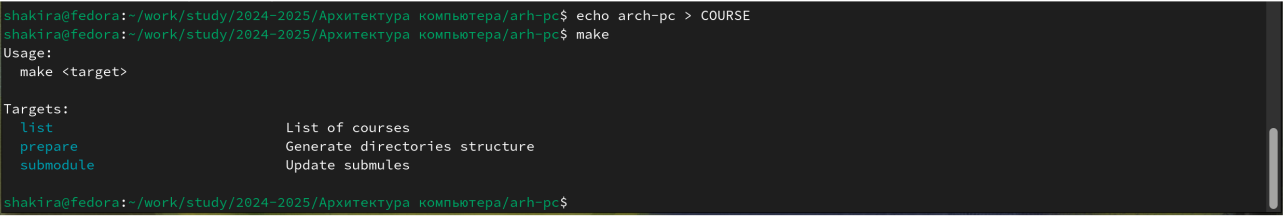
Перехожу в каталог arh-pc с помощью утилиты cd (рис. 18).

Рис. 18: Переход в каталог

Удаляю лишние файлы с помощью утилиты rm (рис. 19).

Рис. 19: Удаление файлов

Создаю необходимые каталоги (рис. 20).

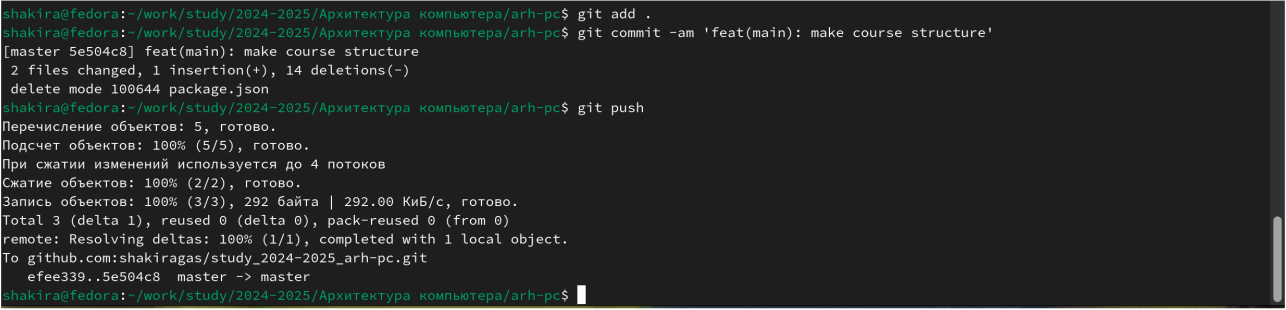
Рис. 20: Создание каталогов

Отправляю созданные каталоги с локального репозитория на сервер:

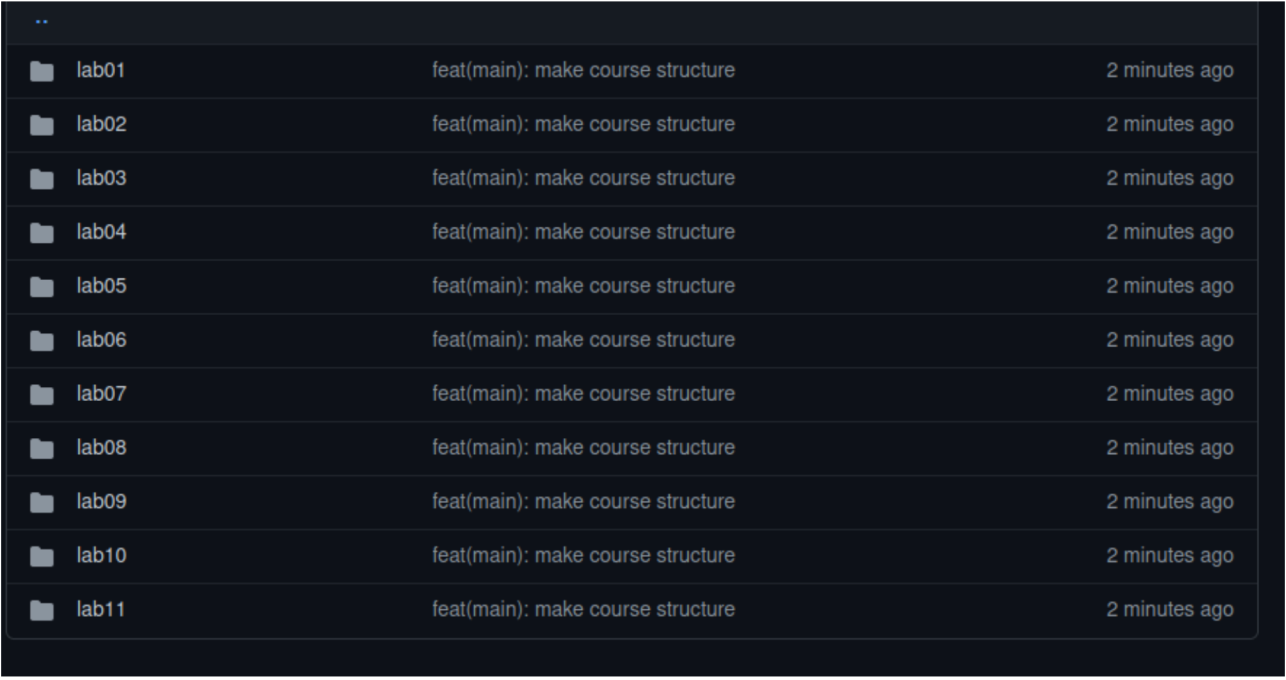
добавляю все созданные каталоги с помощью git add, комментирую и сохраняю

изменения на сервере как добавление курса с помощью git commit и отправляю

всё на сервер с помощью git push (рис. 21).

Рис. 21: Добавление и сохранение изменений на сервере и оправка

Проверяю правильность выполнения работы на сайте GitHub (рис. 22).

Рис. 22: Результат

4.7 Выполнение заданий для самостоятельной работы

1 Задание. Перехожу в директорию labs/lab02/report с помощью утилиты

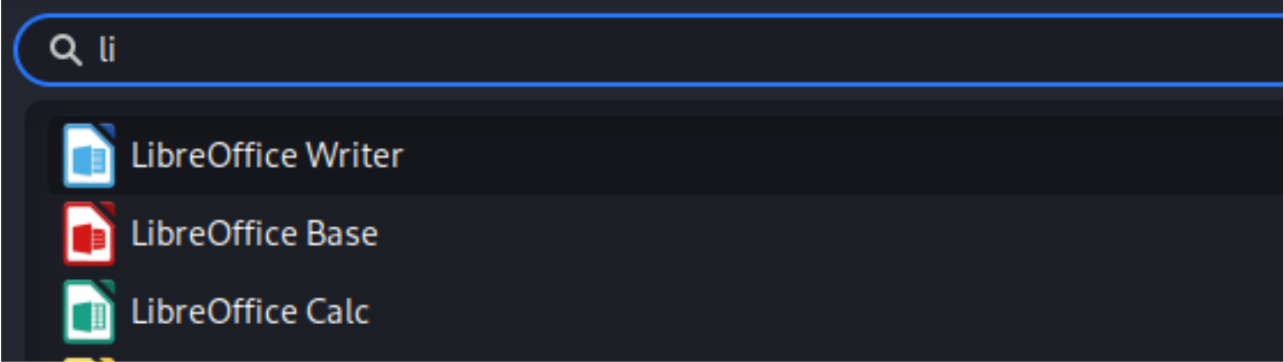
cd. Создаю в каталоге файл для отчета по второй лабораторной работе

с помощью утилиты touch (рис. 23).

Рис. 23: Создание файла

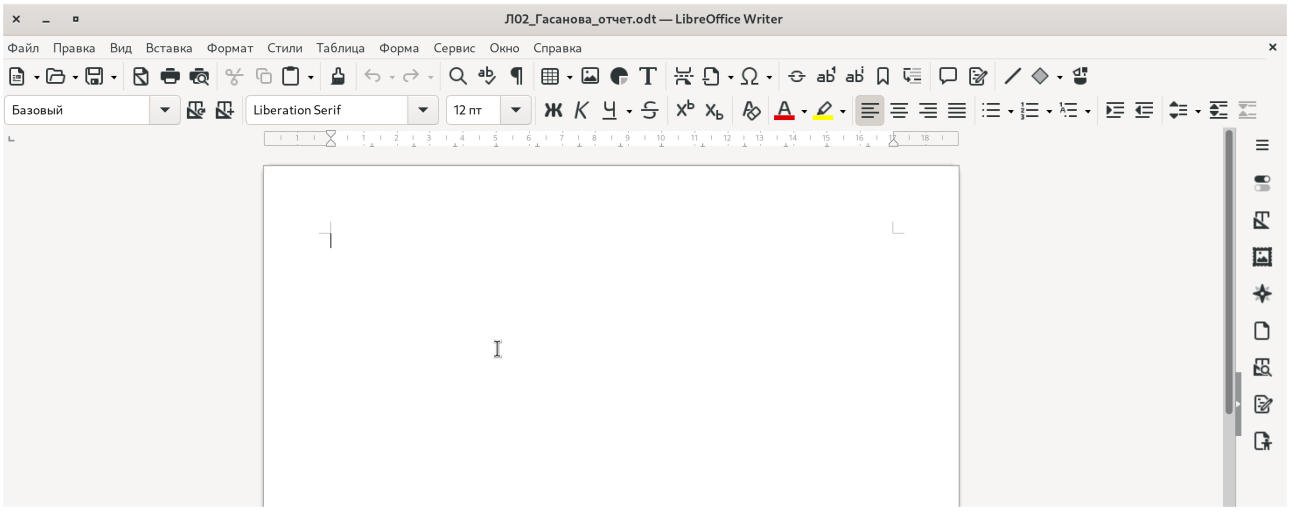
Оформить отчет я смогу в текстовом процессоре LibreOffice Writer, найдя

его в меню приложений (рис. 24).

Рис. 24: Поиск приложений

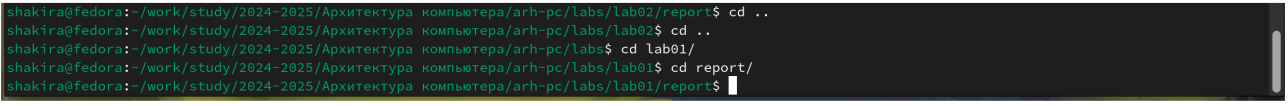
После открытия текстового процессора открываю в нем созданный файл и

могу начать в нем работу над отчетом (рис. 25).

Рис. 25: Текстовый редактор

2 Задание. Перехожу из подкаталога lab02/report в подкаталог lab01/report

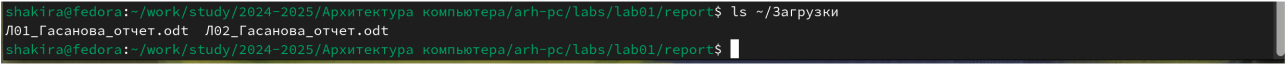
с помощью утилиты cd (рис. 26).

Рис. 26: Перемещение между директориями

Проверяю расположение файлов с отчетами по первой и второй

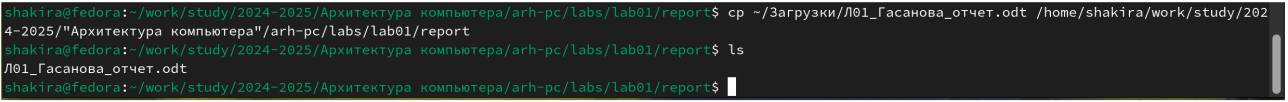
лабораторным работам. Они должны находиться в подкаталоге «Загрузки» в

домашней директории. Для проверки использую команду ls (рис. 27).

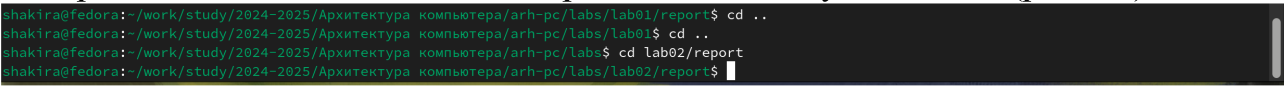
Рис. 27: Проверка

Копирую первую лабораторную с помощью утилиты cp и проверяю

правильность выполнения команды cp с помощью ls (рис. 28).

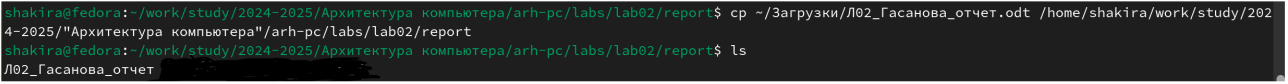
Рис. 28: Копирование первой лабораторной работы

Возвращаюсь в подкаталог lab02/report с помощью утилиты cd (рис. 29).

Рис. 29: Перемещение между директориями

Копирую её с помощью утилиты cp и проверяю правильность выполнения

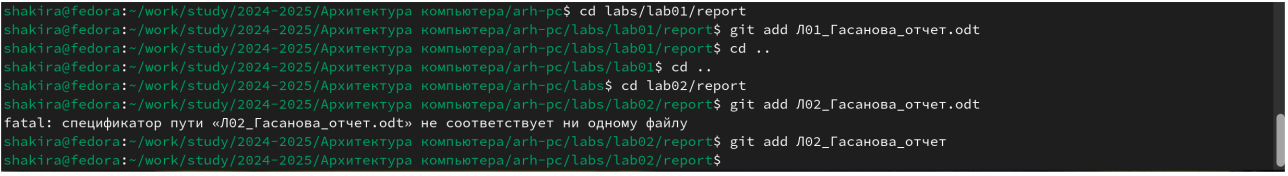
команды с помощью ls (рис. 30).

Рис. 30: Копирование второй лабораторной работы

3 Задание. Добавляю с помощью команды git add в коммит созданные файлы:

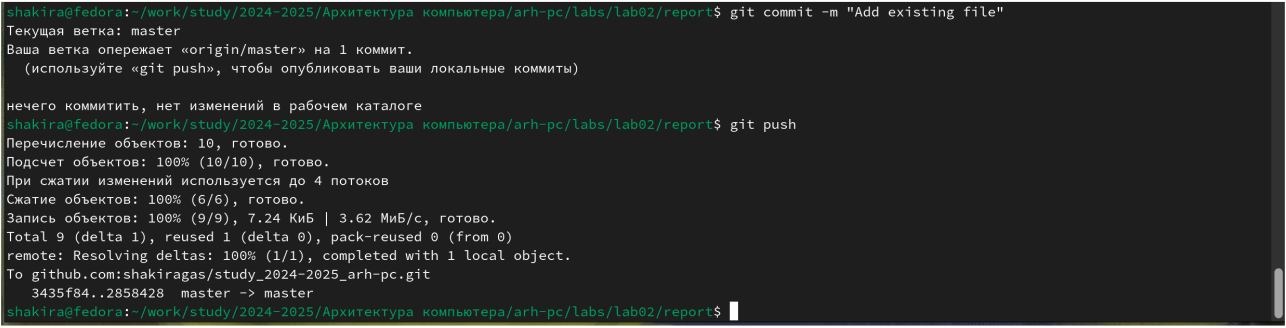
Л01\_Гасанова\_отчет и Л02\_Гасанова\_отчет. Не забываю перемещаться по

директориям, чтобы найти нужные файлы (рис. 31).

Рис. 31: Добавление файлов на сервер

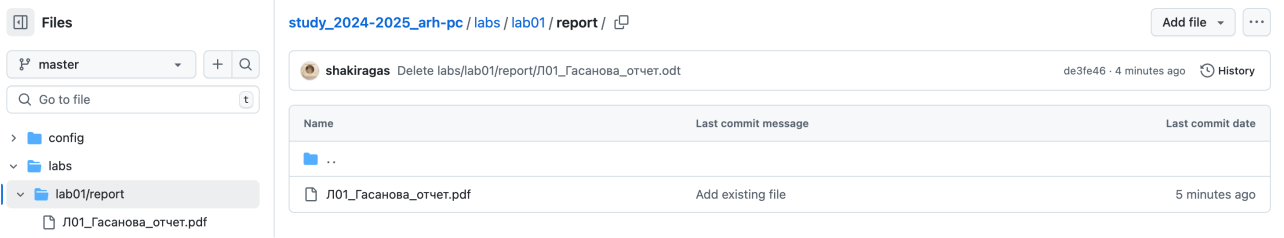
Сохраняю изменения на сервере командой git commit -m “…” и заодно

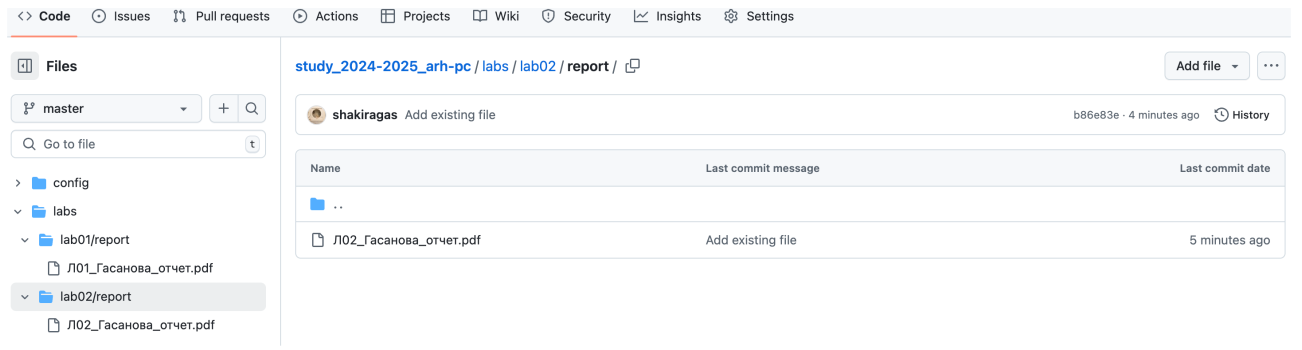
добавляю пояснение (рис. 32).

Рис. 32: Сохранение изменений на сервере

Проверяю на сайте GitHub правильность выполнения заданий и проверяю

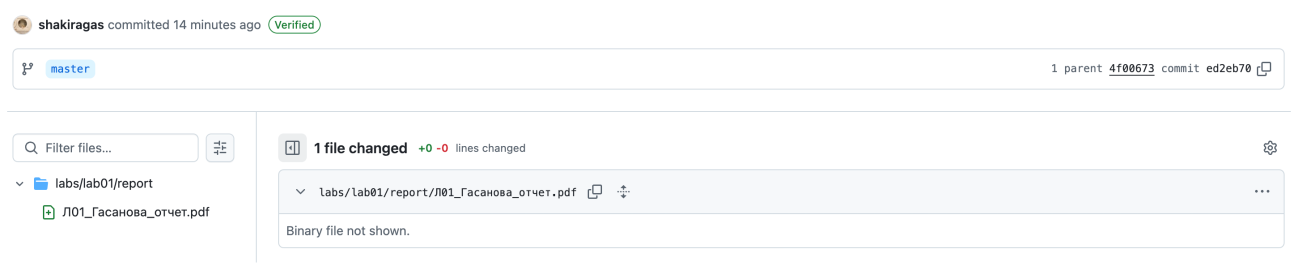
наличие пояснений (рис. 33, рис. 34).

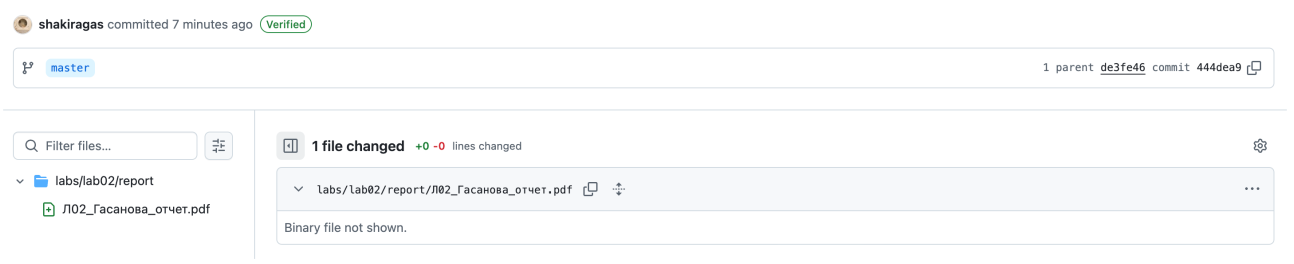
Рис. 33: Проверка на сайте

Рис. 34: Проверка на сайте

При просмотре изменений замечаю, что были добавлены файлы с

отчетами по лабораторным работам (рис. 35, рис. 36).

Рис. 35: Проверка на сайте

Рис. 36: Проверка на сайте

# 5 Выводы

В ходе выполнения этой лабораторной работы я изучила принципы и

применение инструментов контроля версий, а также приобрела практические

навыки работы с системой Git.

# Список литературы

1. Архитектура ЭВМ (rudn.ru)