**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра фундаментальная информатика и**

**информационные технологии**

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2**

*дисциплина: Архитектура компьютеров*

Студент: Волобуев И. А.

Группа: НБИбд-01-23

**МОСКВА**

2023 г.

## Содержание

[1 Цель работы 3](#_Toc12766)

[2 Задание 4](#_Toc12767)

[3 Теоретическое введение 5](#_Toc12768)

[4 Выполнение лабораторной работы 6](#_Toc12769)

[5 Выводы 18](#_Toc12770)

[6 Источники 19](#_Toc12772)

# 1 Цель работы

Целью работы является изучить идеологию и применение средств контроля версий. 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 курс на основе шаблона.
6. Настройка каталога курса.
7. Задания 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 (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. Системы контроля версий также могут обеспечивать дополнительные, более гибкие функ- циональные 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. Кроме того, 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 Gitпредставляет 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.

# 4 Выполнение лабораторной работы

### 1. Настройка Github

Создаю учётную запись на сайте Github и заполняю основные данные (рис. 1)

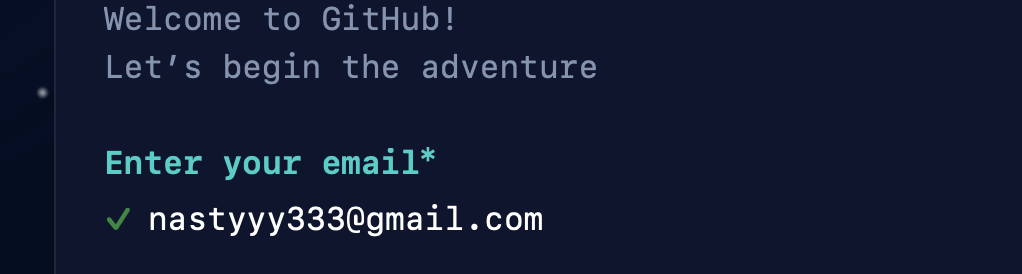


Рис. 1. Заполнение данных учётной записи Github

### 2. Базовая настройа Git

Сначала делаю предварительную конфигурацию Git. Открываю терминал и ввожу команды указав имя и email. (рис. 2).

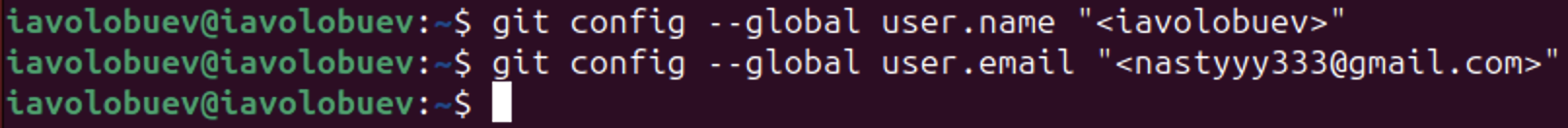


Рис. 2. Предварительная конфигурация Git

Настраиваю utf-8 в выводе сообщений Git (рис.13).



Рис. 3. Настройка кодировки

Задаю имя ‘master’ для начальной ветки(рис. 4).



Рис. 4. Создание имени для начальной ветки

Задаю параметр autocrlf (рис. 5).



Рис. 5. Параметр autocrlf

Задаю параметр safecrlf (рис. 6).



Рис. 6. Параметр safecrlf

### 3. Создание SSH ключа

Для последующей идентификации пользователя на сервере репозиториев необходимо сгенерировать пару ключей (приватный и открытый). Для этого использую команду ssh-keygen -C ‘Имя пользователя , <work@mail>’. Ключ автоматически сохранится в каталоге ~/.ssh/(рис. 7).

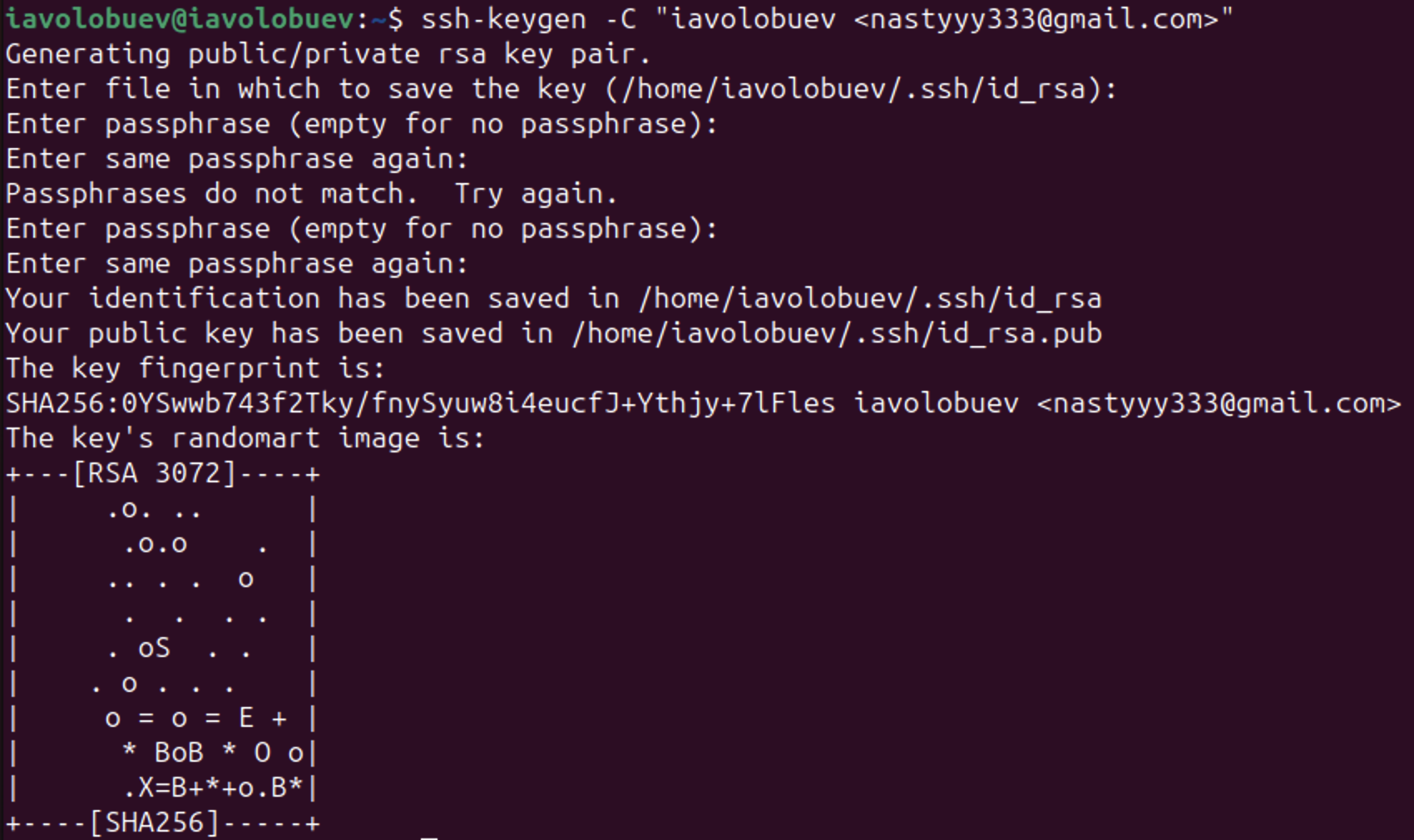


Рис. 7. Генерация SSH ключа

Читаю содержимое файла при помощи команды cat и копирую(рис. 8).

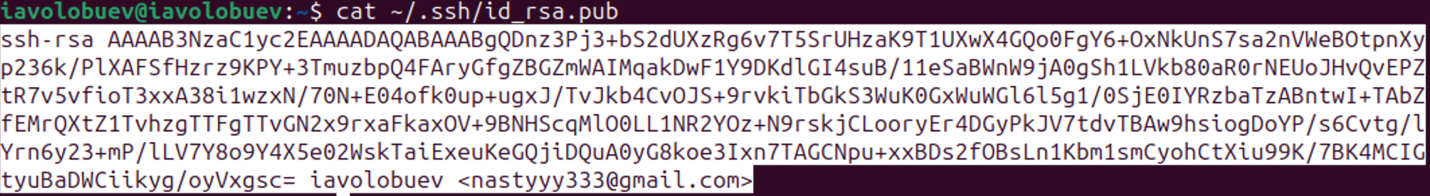


Рис. 8. Чтение файла

Открываю браузер, захожу на сайтт Github. Открываю свой профиль и выбираю страницу “SSH and GPG keys”. Нажимаю кнопку «New SSH key». Вставляю скопированный ключ в поле «Key». В поле «Title» указываю имя для ключа. Нажимаю «Add SSH key», чтобы завершить добавление ключа (рис. 9).

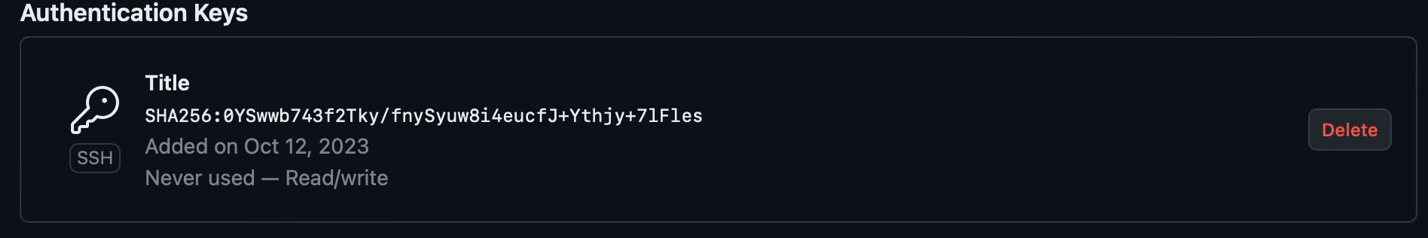


Рис. 9. Добавление ключа

### 4. Создание рабочего пространства и репозитория курса на основе шаблона

Перехожу в терминал. Создаю директорию, рабочее пространство , с помощью утилиты mkdir , далее при помощи ключа -p создаю все директории после домашней. (рис. 10).



Рис. 10. Создание рабочего пространства

### 5. Создание репозитория курса на основе шаблона

Открываю браузер и перехожу на страницу репозитория с шаблоном курса. Далее выбираю «Use this template», чтобы использовать этот шаблон (рис. 11).

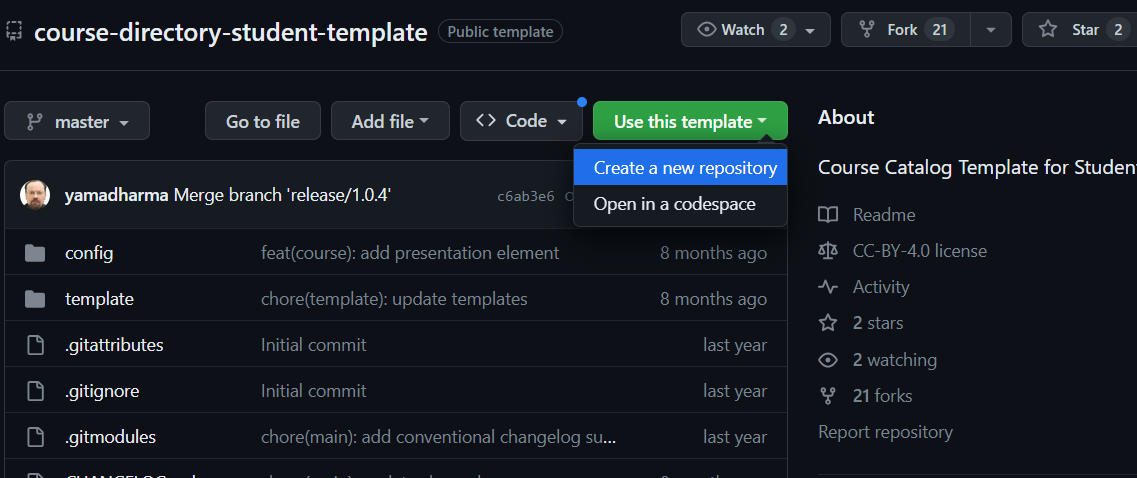


Рис. 11.Страница шаблона для репозитория

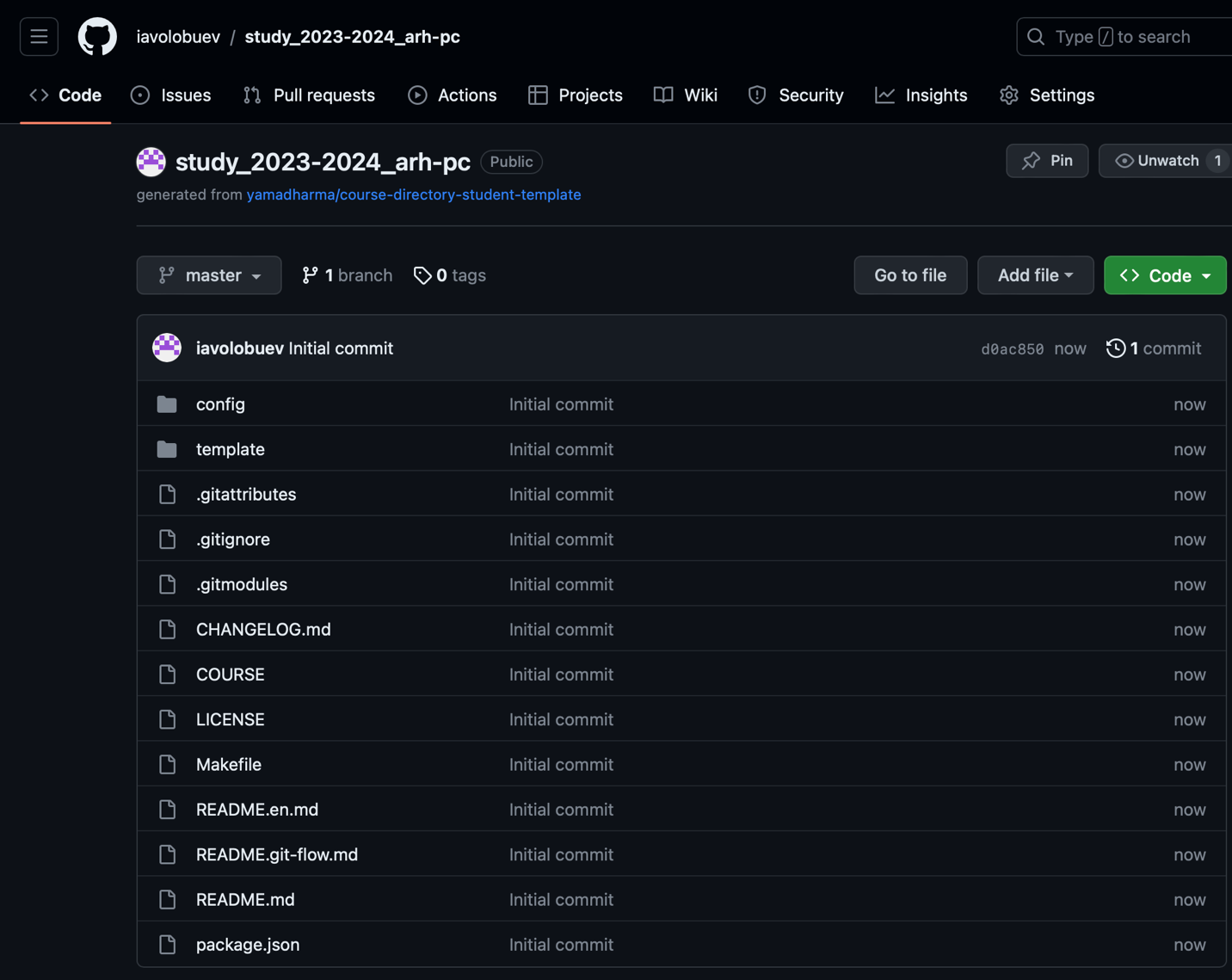
Потом задаю именя репозитория(study\_2023-2024\_arch-pc) и создаю репозиторий(рис. 12) 

Рис. 12. Созданный репозиторий

Через терминал перехожу в созданный каталог курса с помощью утилиты cd(рис. 13).



Рис. 13. Перемещение между директориями

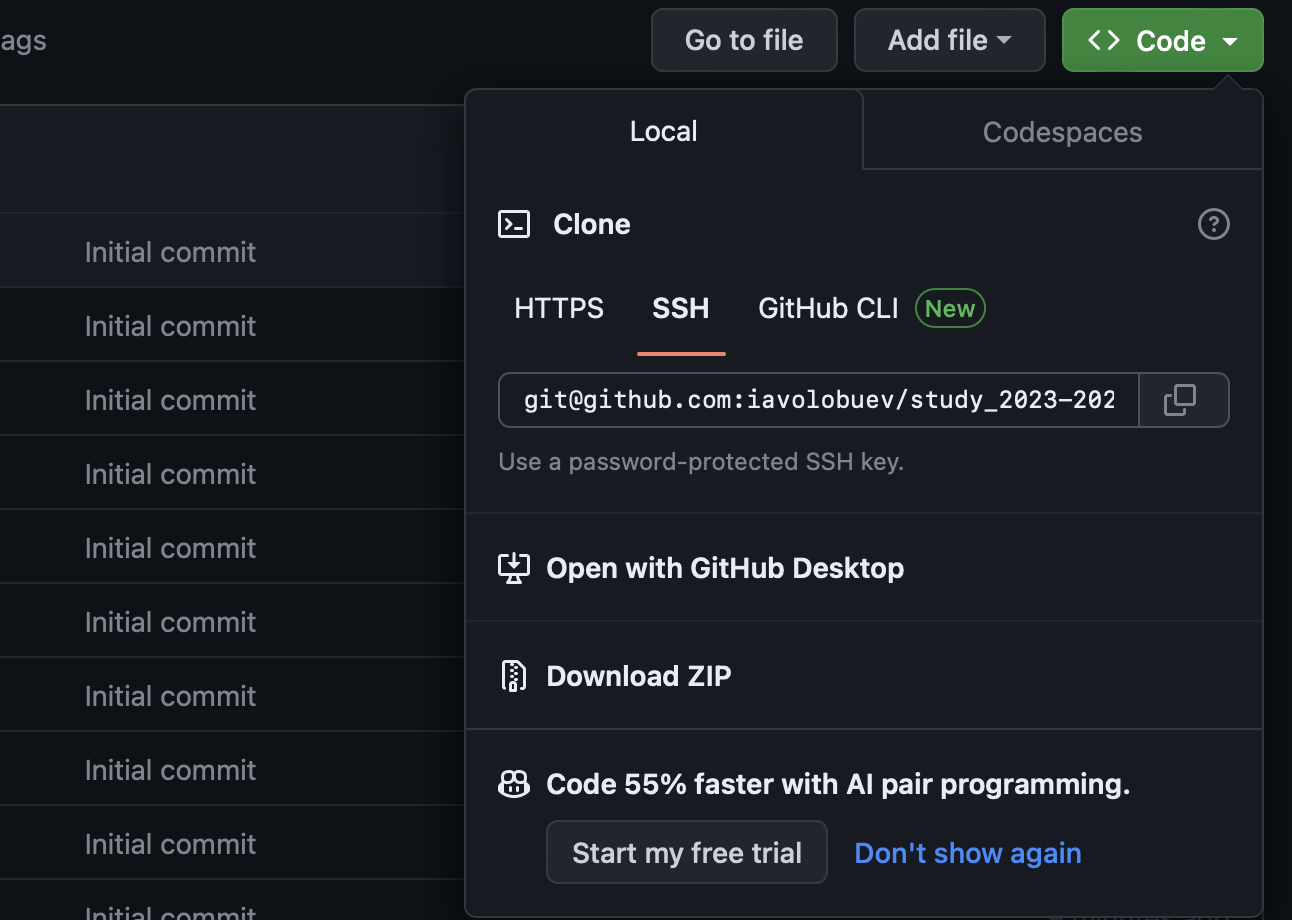
Копирую ссылку для клонирования на странице созданного репозитория, нажав на «Code», потом на «SSH»(рис. 14). 

Рис. 14. Окно с ссылкой для копирования репозитория

Клонирую созданный репозиторий с помощью команды git clone --recursive(рис.15).



Рис. 15. Клонирование репозитория

**6.Настройка каталога курса**

Перехожу в каталог arch-pc при помощи утилиты cd(рис. 16). 

Рис. 16. Перемещение между директориями

Удаляю лишние файлы при помощи утилиты rm(рис. 17).



Рис. 17. Удаление файлов

Создаю необходимые каталоги(рис. 18).



Рис. 18. Создание каталогов

Отправляю файлы на сервер. При помощи команды git add добавляю все созданные каталоги и комментирую при помощи команды git commit(рис. 19).

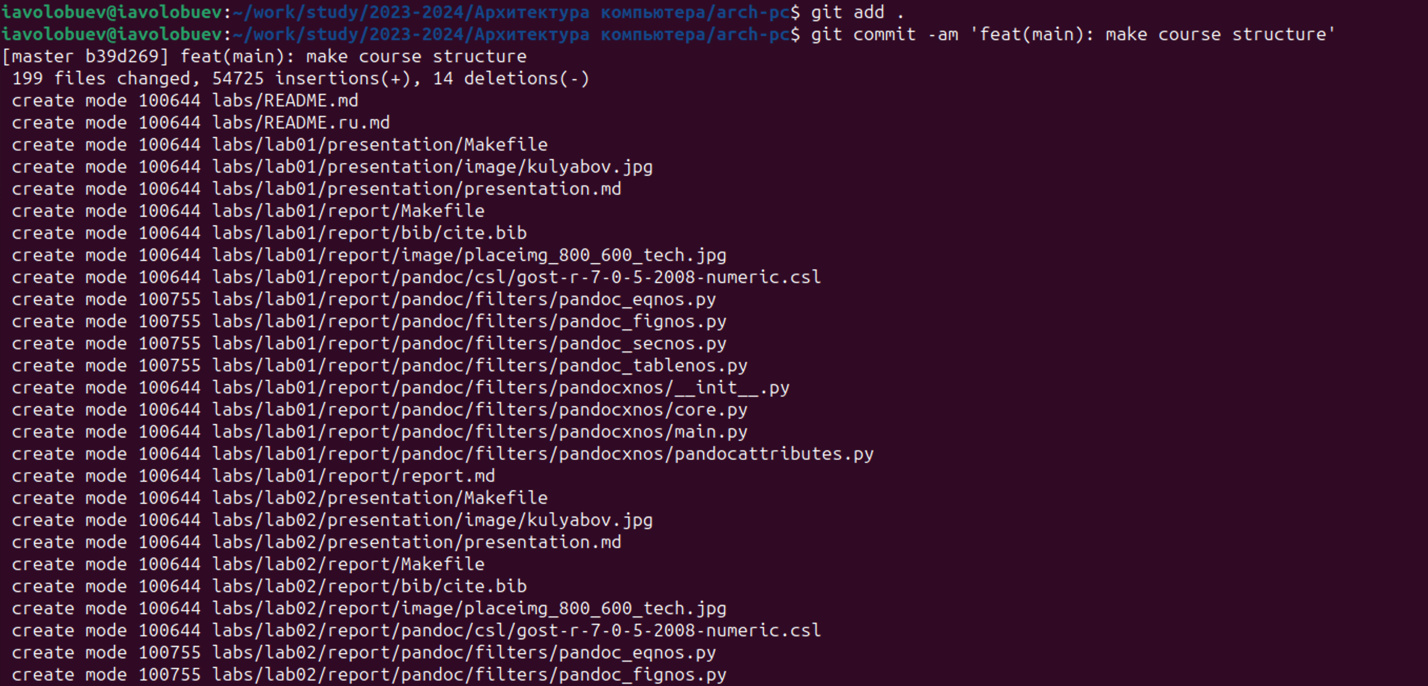


Рис. 19. Добавление и сохранение изменений на сервере

Отправляю на сервер с помощью команды git push(рис. 20).

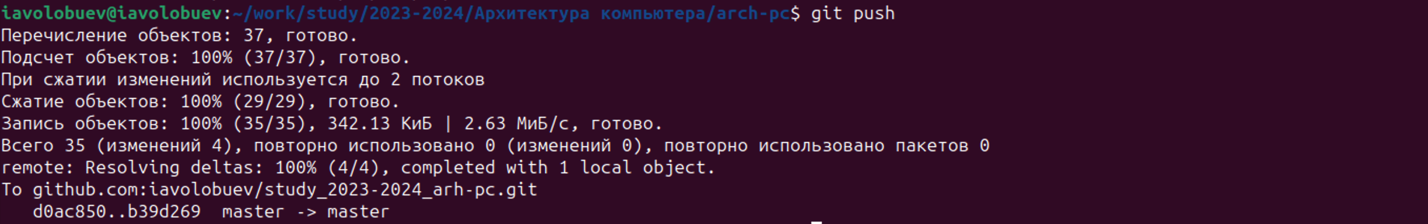


Рис. 20. Выгрузка изменений на сервер

Проверяю правильность выполнения на сайте Github(рис. 21).

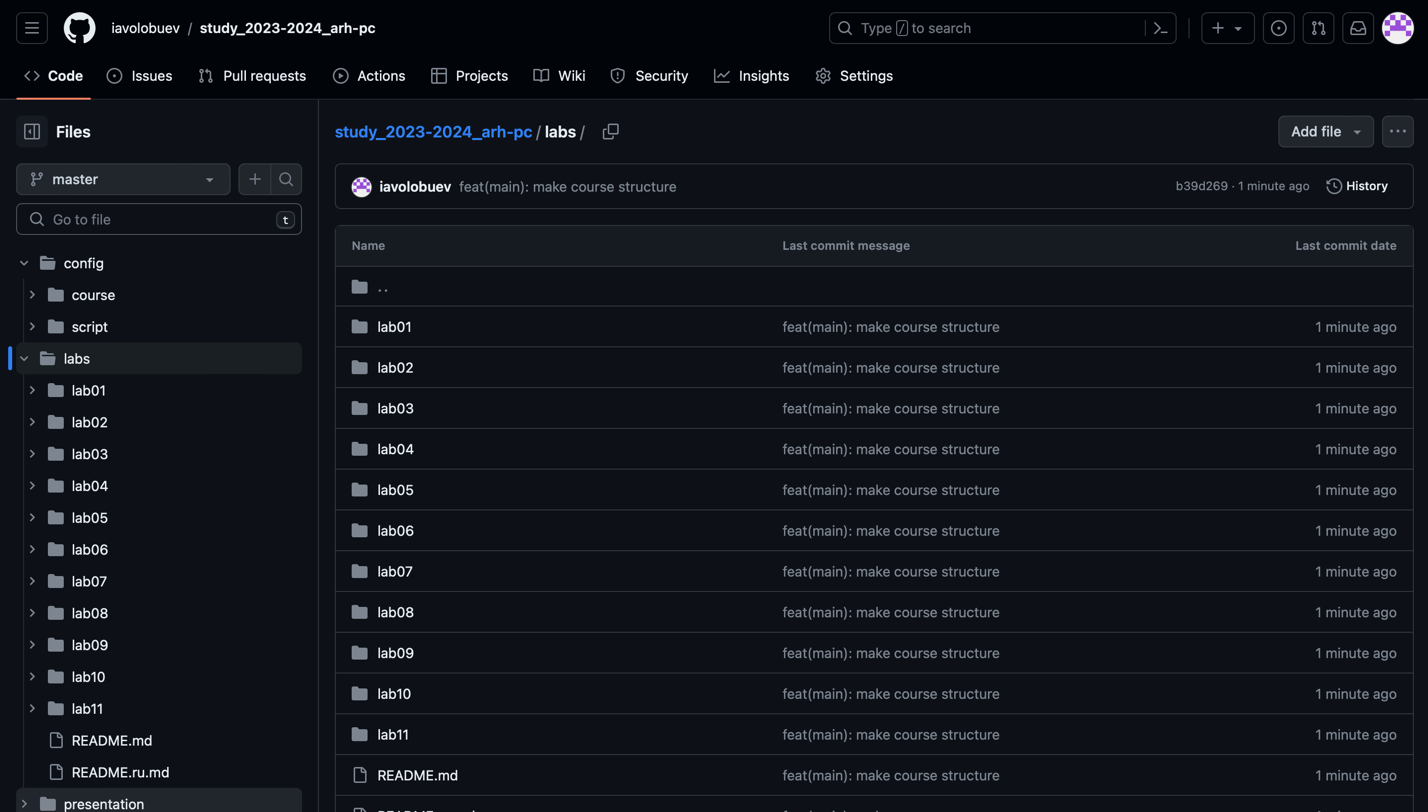


Рис. 21. Страница репозитория

**7.Выполнение заданий для самостоятельной работы**

Перехожу в директорию labs/lab02/report c помощью утилиты cd. Создаю в каталоге файл для отчёта по второй лабораторной работе с помощью утилиты touch (рис.22).



Рис.22. Создание файла

Оформить отчёт я могу в текством редакторе LibreOffice Writer(рис.23).

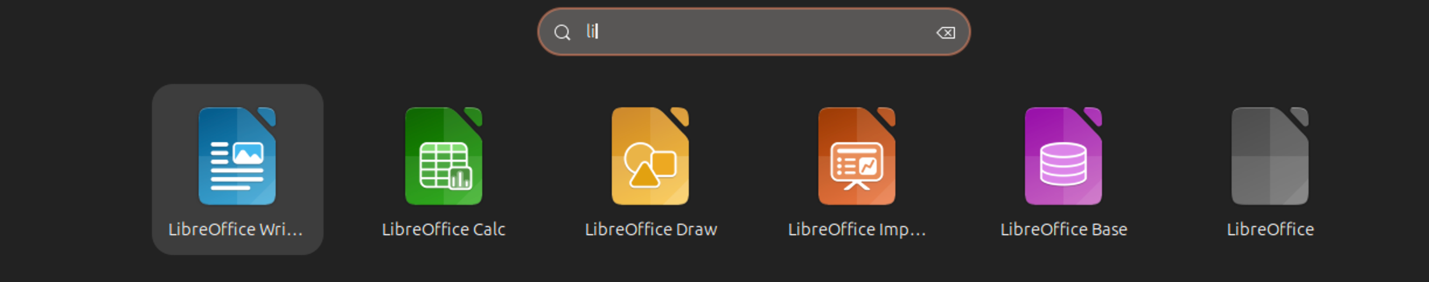


Рис. 23. Меню приложений

Открываю в текстом редакторе файл , где буду делать отчёт(рис.24).

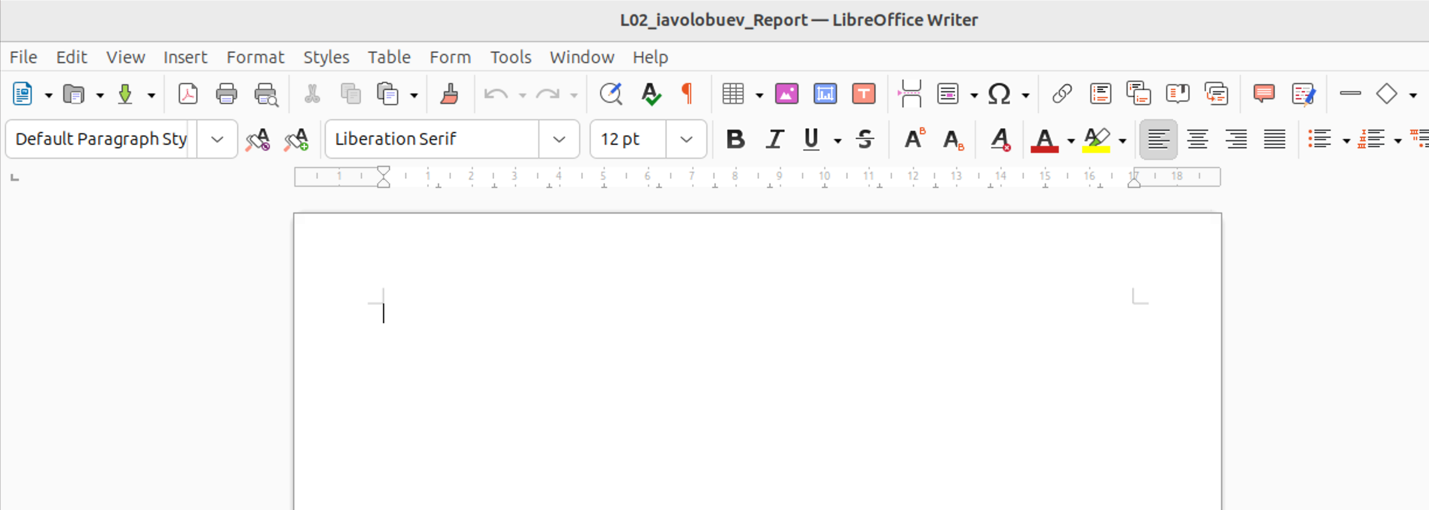


Рис. 24. Текстовый редактор

Перехожу в подкатолог lab01/report c помощью утилиты cd(рис.25).



Рис. 25. Перемещение между директориями

Проверяю где находится файл с отчётом по первой лабораторной работе(рис.26).

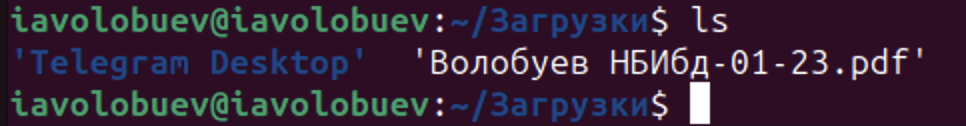


Рис.26. Проверка нахождения файла

Копирую первую лабораторную с помощью утилиты cp и проверяю правильность выполнения команды(рис.27.).

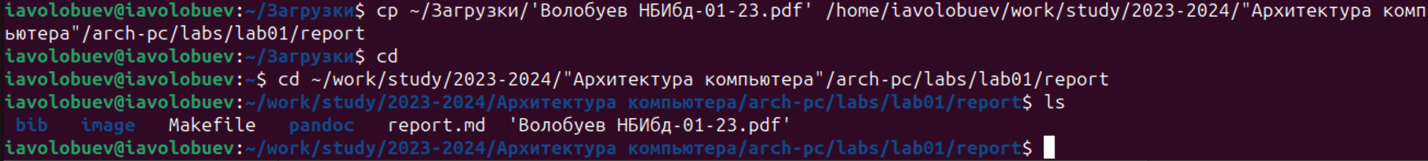


Рис.27. Копирование файла

Добавляю файл «Волобуев НБИбд-01-23» при помощи команды git add и сохраняю изменения при помощи git commit (рис.28,29).



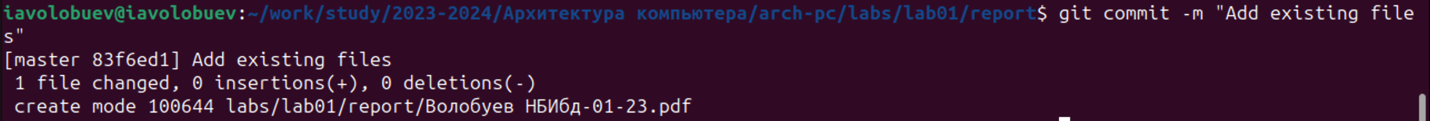


Рис.28,29. Добавление и сохранения файла

Проверяю правильность выполнения задания(рис.30)

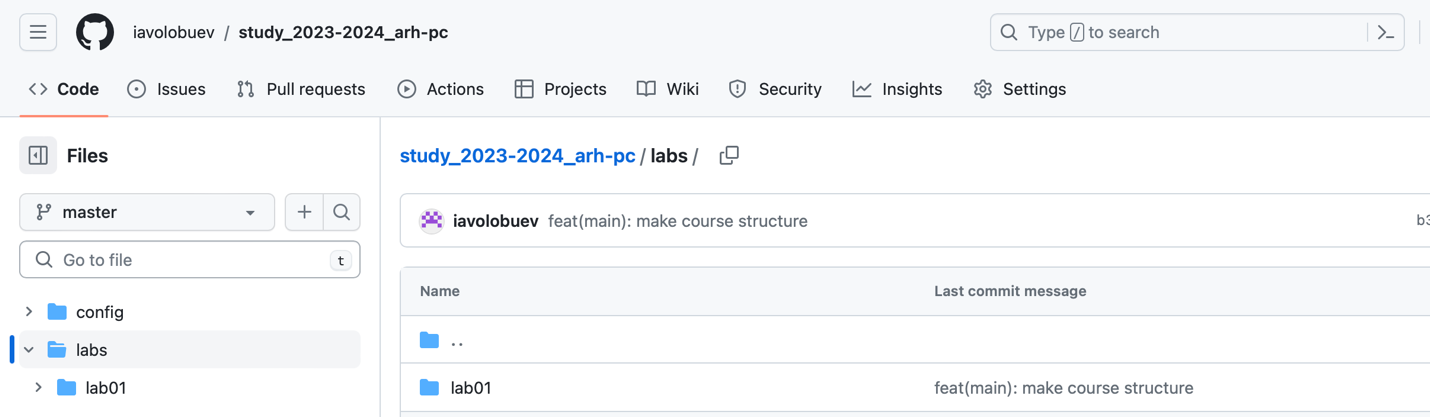


Рис.30.Страница каталога в репозитории.

# 

# 5 Выводы

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

# 6 Источники

1. [Архитектура ЭВМ (rudn.ru)](https://esystem.rudn.ru/pluginfile.php/1584618/mod_resource/content/1/%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.pdf)