

Лабораторная работа №2

Система контроля версий Git

Коровкин Никита Михайлович

Содержание

| | | |
|----------|--|-----------|
| 1 | Цель работы | 5 |
| 2 | Выполнение лабораторной работы | 6 |
| 3 | Выполнение заданий для самостоятельной работы | 16 |
| 4 | Выводы | 20 |

Список иллюстраций

| | | |
|------|--|----|
| 2.1 | Установка git | 6 |
| 2.2 | Настройка имени и адреса email | 6 |
| 2.3 | Настройка utf-8 | 6 |
| 2.4 | Добавление названия начальной ветке | 7 |
| 2.5 | Параметры autocrlf и safecrlf | 7 |
| 2.6 | Генерация SSH ключа | 8 |
| 2.7 | Меню GitHub и раздел settings | 9 |
| 2.8 | Раздел для создания SSH ключа | 10 |
| 2.9 | SSH ключ | 10 |
| 2.10 | Использование команды для копирования в буфер обмена | 10 |
| 2.11 | Создание каталога “Архитектура компьютера” | 10 |
| 2.12 | Используем шаблон для создания своего репозитория | 11 |
| 2.13 | Созданный репозиторий | 11 |
| 2.14 | Клонирование репозитория | 11 |
| 2.15 | Весь процесс клонирования репозитория | 12 |
| 2.16 | Переход в нужный каталог. | 12 |
| 2.17 | Удаление лишних файлов | 12 |
| 2.18 | Создание новых файлов | 13 |
| 2.19 | Добавление каталогов на сервер, сохранение изменений | 13 |
| 2.20 | Загрузка изменений на сервер | 13 |
| 2.21 | Файлы GitHub | 14 |
| 2.22 | Файлы на компьютере | 15 |
| 3.1 | Создание отчёта | 16 |
| 3.2 | Сохранение файла в нужном каталоге | 17 |
| 3.3 | Копируем файл в нужную директорию | 17 |
| 3.4 | Использование команд add и commit | 18 |
| 3.5 | Загрузка файлов на сервер | 18 |
| 3.6 | Загруженные каталоги с файлами | 19 |

Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

Перед началом работы с git нам необходимо сперва установить его, а затем сделать предварительную конфигурацию (см. рис. 1):

```
[liveuser@localhost-live ~]$ dnf install git
Error: This command has to be run with superuser privileges (under the root user on most systems)
.
[liveuser@localhost-live ~]$ sudo dnf install git
Fedora 38 - aarch64                                1.2 MB/s | 79 MB      01:04
Fedora 38 openh264 (From Cisco) - aarch64          1.2 kB/s | 2.6 kB     00:02
Fedora Modular 38 - aarch64                        532 kB/s | 2.7 MB     00:05
Fedora 38 - aarch64 - Updates                      836 kB/s | 41 MB      00:49
Fedora Modular 38 - aarch64 - Updates              514 kB/s | 2.1 MB     00:04
Package git-2.40.0-1.fc38.aarch64 is already installed.
Dependencies resolved.
Nothing to do.
Complete!
[liveuser@localhost-live ~]$
```

Рис. 2.1: Установка git

Откроем терминал и введём первые две команды. В них укажем имя и email владельца репозитория (см. рис. 2).

```
[liveuser@localhost-live ~]$ git config --global user.name "Nikita Korovkin"
[liveuser@localhost-live ~]$ git config --global user.email "zane41181@gmail.com"
[liveuser@localhost-live ~]$
```

Рис. 2.2: Настройка имени и адреса email

После того, как мы задали имя пользователя и адрес электронной почты, введём команду, чтобы настроить utf-8 в выводе сообщений git (см. рис. 3).

```
[liveuser@localhost-live ~]$ git config --global core.quotePath false
[liveuser@localhost-live ~]$
```

Рис. 2.3: Настройка utf-8

Теперь мы зададим имя начальной ветки. Она будет называться master (см. рис. 4).

```
[liveuser@localhost-live ~]$ git config --global init.defaultBranch master
```

Рис. 2.4: Добавление названия начальной ветке

Затем зададим следующие два параметра: autocrlf и safecrlf (см. рис. 5). Параметр autocrlf нужен для того, чтобы в главном репозитории все переводы строк в текстовых файлах были одинаковы. А команда safecrlf проверяет обратимость преобразования для текущей настройки.

```
[liveuser@localhost-live ~]$ git config --global core.autocrlf input  
[liveuser@localhost-live ~]$ git config --global core.safecrlf warn
```

Рис. 2.5: Параметры autocrlf и safecrlf

Чтобы продолжить работу, нам необходимо сперва сгенерировать открытый SSH ключ. Он необходим для идентификации пользователя на сервере репозитория. В качестве аргументов необходимо указать свое имя и электронную почту(см.рис. 6).

```
[liveuser@localhost-live ~]$ ssh-keygen -C "Nikita Korovkin zane41181@gmail.com"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/liveuser/.ssh/id_rsa):
Created directory '/home/liveuser/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/liveuser/.ssh/id_rsa
Your public key has been saved in /home/liveuser/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:M4zd8dwt0u1igFn6RJnnRfom4UyTnp1E/g5QZs0ogxc Nikita Korovkin zane41181@gmail.com
The key's randomart image is:
+---[RSA 3072]-----+
|      o=..|
|      E .*+= |
|      ..o* Bo.|
|      +..+X @.Xo|
|      . S.=.* %.B|
|      o o o B |
|      . o o |
|      . . |
|      |
+----[SHA256]-----+
[liveuser@localhost-live ~]$
```

Рис. 2.6: Генерация SSH ключа

Теперь вставим этот ключ в специальное поле на сайте GitHub. Оно находится в разделе settings. Перейдя в settings, найдём пункт SSH и GPS keys. Нажимаем на SSH key, вставляем ключ в нужное поле(см.рис.7-9).



Pancakeboy1987



Set status



Your profile



Your repositories



Your Copilot



Your projects



Your stars



Your gists



Your organizations



Your enterprises



Your sponsors



Try Enterprise



Feature preview



Settings

Рис. 2.7: Меню GitHub и раздел settings

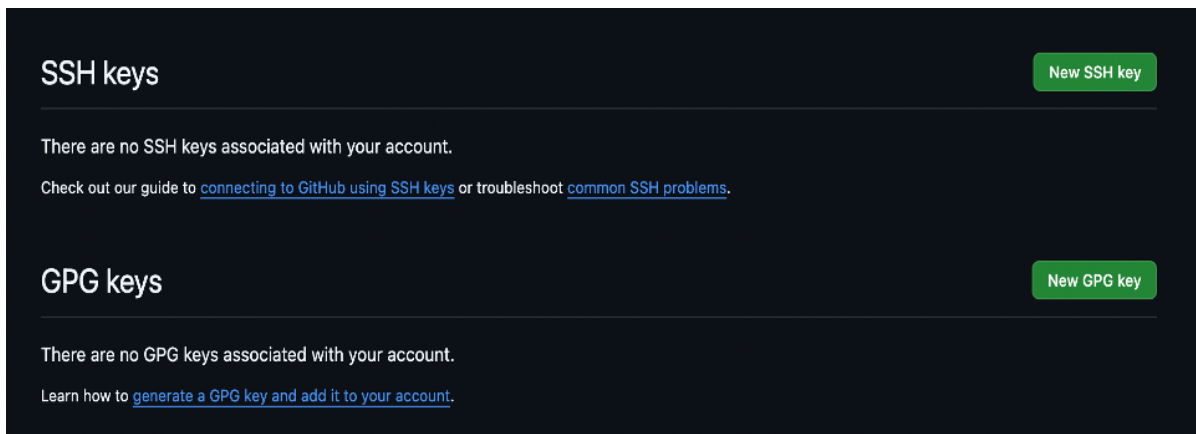


Рис. 2.8: Раздел для создания SSH ключа

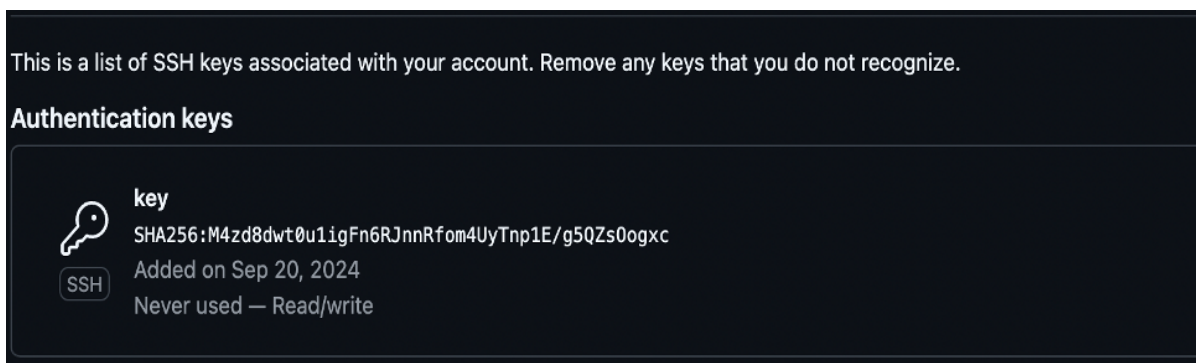


Рис. 2.9: SSH ключ

Сгенерированный SSH ключ мы копируем в буфер обмена при помощи команды `cat ~/.ssh/id_rsa.pub | xclip -sel clip` (см.рис.10).

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

Рис. 2.10: Использование команды для копирования в буфер обмена

После того, как мы добавили ключ нам необходимо при помощи терминала создать каталог «Архитектура компьютера» (см.рис.11).

```
[liveuser@localhost-live ~]$ mkdir -p ~/work/study/2024-2025/"Архитектура компьютера"
```

Рис. 2.11: Создание каталога «Архитектура компьютера»

Теперь на сайте GitHub нам нужно будет создать репозиторий курса на основе шаблона. Шаблон находится по данному адресу: <https://github.com/yamadharma/cour>

se-directory-student-template. Перейдя по ссылке, нажимаем «use this template» и создаём свой репозиторий(см.рис.12-13).

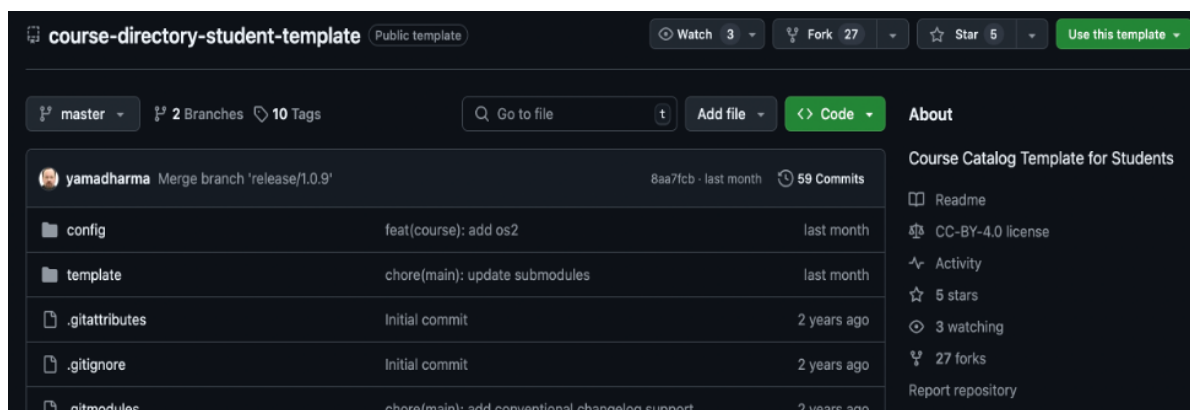


Рис. 2.12: Используем шаблон для создания своего репозитория

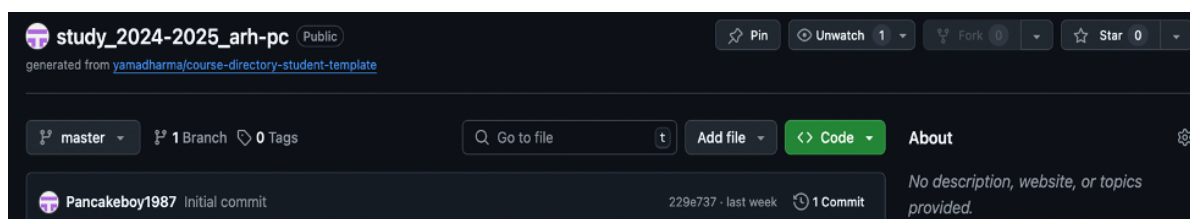


Рис. 2.13: Созданный репозиторий

Теперь, когда репозиторий создан на сайте, нам необходимо клонировать его. Для этого воспользуемся приведенной ниже командой(см.рис.14).

```
[liveuser@localhost-live ~]$ git clone --recursive git@github.com:Pancakeboy1987/study_2024-2025_arh-pc.git
```

Рис. 2.14: Клонирование репозитория

```

Cloning into 'study_2024-2025_arh-pc'...
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (32/32), done.
remote: Total 33 (delta 1), reused 18 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (33/33), 18.81 KiB | 4.70 MiB/s, done.
Resolving deltas: 100% (1/1), done.
Submodule 'template/presentation' (https://github.com/yamadharma/academic-presentation-markdown-template.git) registered for path 'template/presentation'
Submodule 'template/report' (https://github.com/yamadharma/academic-laboratory-report-template.git) registered for path 'template/report'
Cloning into '/home/liveuser/study_2024-2025_arh-pc/template/presentation'...
remote: Enumerating objects: 111, done.
remote: Counting objects: 100% (111/111), done.
remote: Compressing objects: 100% (77/77), done.
remote: Total 111 (delta 42), reused 100 (delta 31), pack-reused 0 (from 0)
Receiving objects: 100% (111/111), 102.17 KiB | 1.00 MiB/s, done.
Resolving deltas: 100% (42/42), done.
Cloning into '/home/liveuser/study_2024-2025_arh-pc/template/report'...
remote: Enumerating objects: 142, done.
remote: Counting objects: 100% (142/142), done.
remote: Compressing objects: 100% (97/97), done.
remote: Total 142 (delta 60), reused 121 (delta 39), pack-reused 0 (from 0)
Receiving objects: 100% (142/142), 341.09 KiB | 2.01 MiB/s, done.
Resolving deltas: 100% (60/60), done.
Submodule path 'template/presentation': checked out 'c9b2712b4b2d431ad5086c9c72a02bd2fca1d4a6'
Submodule path 'template/report': checked out 'c26e22effe7b3e0495707d82ef561ab185f5c748'
[liveuser@localhost-live ~]$

```

Рис. 2.15: Весь процесс клонирования репозитория

На рисунке выше(см.рис.15) изображен весь процесс клонирования. После того, как процесс завершен, можно перейти к следующему этапу – настройке каталога курса.

Для этого сперва перейдем в нужный каталог с помощью `cd ~/work/study/2024-2025/“Архитектура компьютера”/arch-pc` (см.рис.16)

```

[liveuser@localhost-live ~]$ cd ~/work/study/2024-2025/"Архитектура компьютера"/arch-pc

```

Рис. 2.16: Переход в нужный каталог.

Теперь удалим ненужные файлы при помощи команды `rm`(см.рис.17)

```

[liveuser@localhost-live study_2024-2025_arh-pc]$ rm package.json
[liveuser@localhost-live study_2024-2025_arh-pc]$

```

Рис. 2.17: Удаление лишних файлов

После того, как все лишние файлы удалены, создадим новые при помощи команд `echo arch-pc > COURSE` и `make`(см.рис.18).

```
[liveuser@localhost-live study_2024-2025_arh-pc]$ echo arch-pc > COURSE
[liveuser@localhost-live study_2024-2025_arh-pc]$ make
```

Рис. 2.18: Создание новых файлов

Следующий этап – отправка файлов на сервер. Для этого мы воспользуемся командой `git add .`, которая предназначена для добавления на сервер сразу всех файлов и каталогов. После этого пропишем команду `git commit -am 'feat(main): make course structure'`. Она сохраняет внесённые изменения и добавляет комментарий (см.рис.19).

```
[liveuser@localhost-live study_2024-2025_arh-pc]$ git add .
[liveuser@localhost-live study_2024-2025_arh-pc]$ git commit -am 'feat(main): make course structure'
```

Рис. 2.19: Добавление каталогов на сервер, сохранение изменений

Теперь воспользуемся командой `push`, чтобы загрузить все изменения на сервер(см.рис.20).

```
[liveuser@localhost-live study_2024-2025_arh-pc]$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 293 bytes | 293.00 KiB/s, done.
Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:Pancakeboy1987/study_2024-2025_arh-pc.git
  229e737..7ea45c6 master -> master
[liveuser@localhost-live study_2024-2025_arh-pc]$
```

Рис. 2.20: Загрузка изменений на сервер

Остаётся только сравнить файлы GitHub с файлами на компьютере (см.рис.21-22).















| | | | |
|---|-----------------------------------|--|----------------|
|  Pancakeboy1987 feat(main): make course stru... 7ea45c6 - 39 minutes ago  2 Commits | | | |
|  config | Initial commit | | last week |
|  template | Initial commit | | last week |
|  .gitattributes | Initial commit | | last week |
|  .gitignore | Initial commit | | last week |
|  .gitmodules | Initial commit | | last week |
|  CHANGELOG.md | Initial commit | | last week |
|  COURSE | feat(main): make course structure | | 39 minutes ago |
|  LICENSE | Initial commit | | last week |
|  Makefile | Initial commit | | last week |
|  README.en.md | Initial commit | | last week |
|  README.git-flow.md | Initial commit | | last week |
|  README.md | Initial commit | | last week |

Рис. 2.21: Файлы GitHub

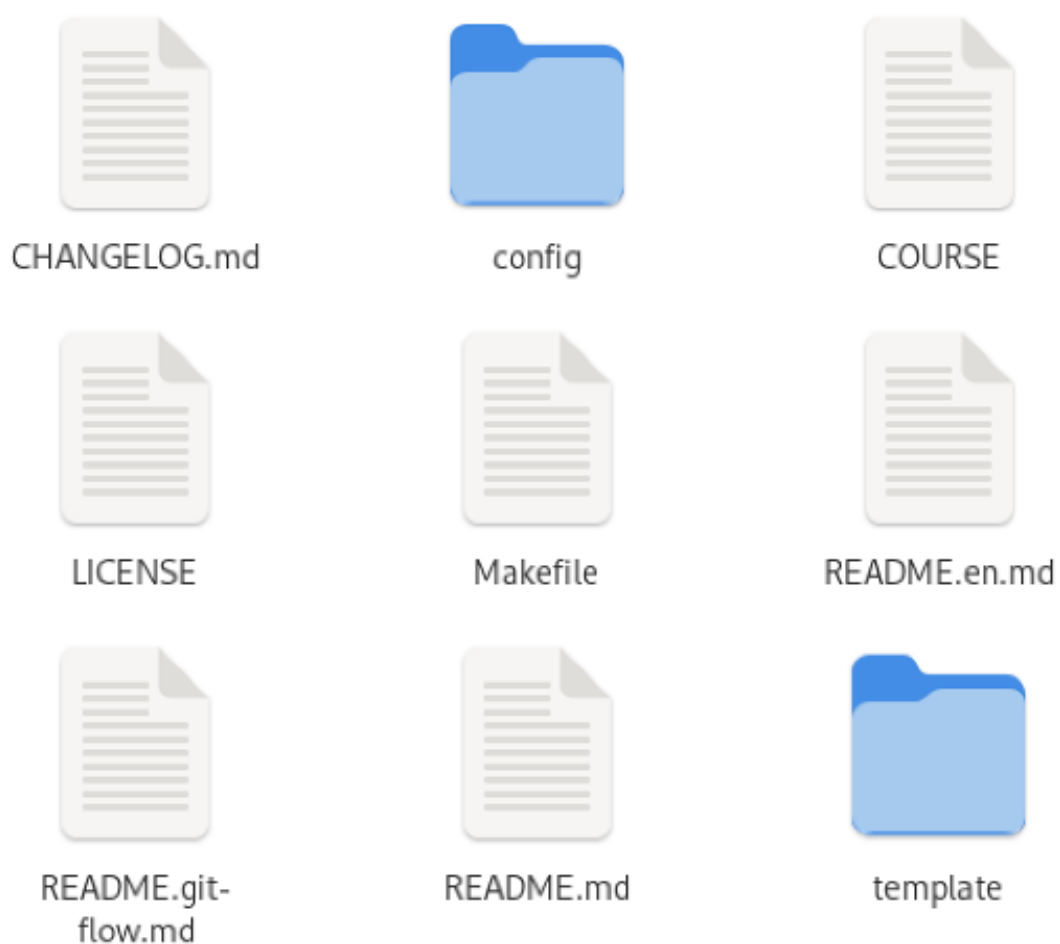


Рис. 2.22: Файлы на компьютере

Файлы совпадают, значит работа выполнена правильно.

3 Выполнение заданий для самостоятельной работы

Приступим к выполнению самостоятельной работы. Для этого сперва создадим файл отчёта. Для этого воспользуемся LibreOffice(см.рис.23).

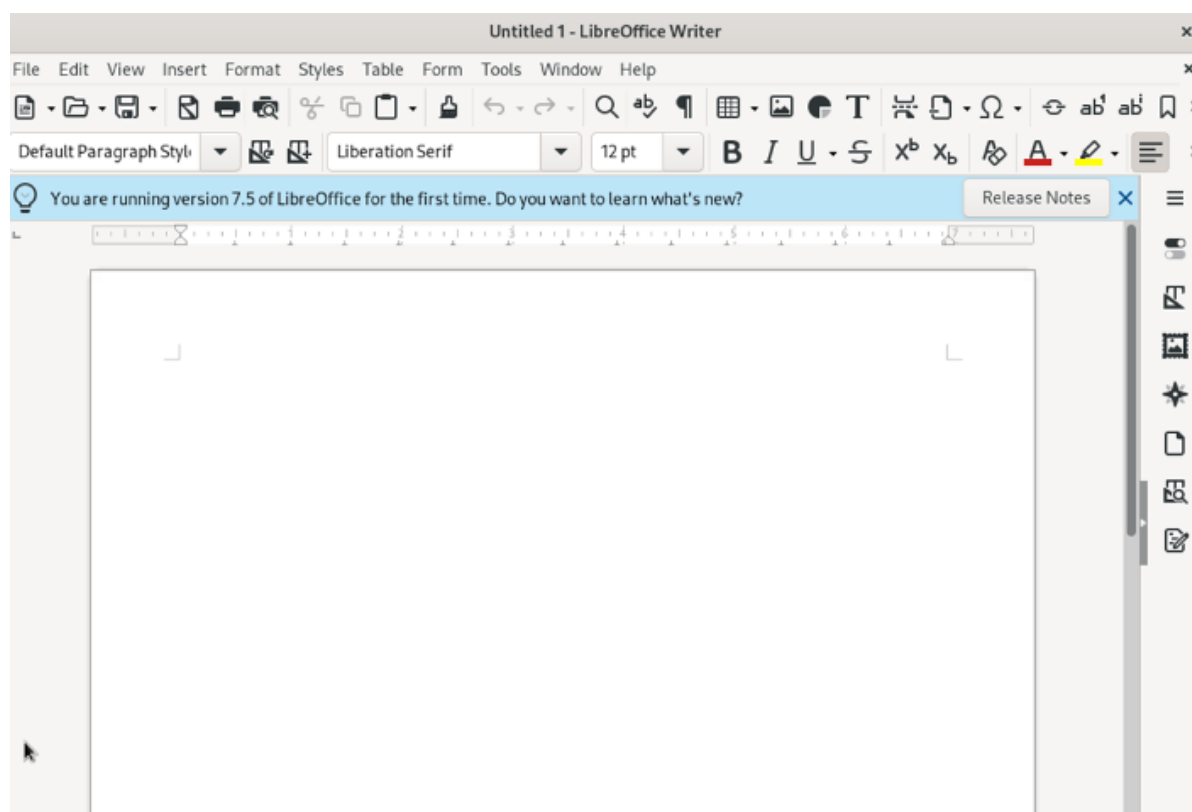


Рис. 3.1: Создание отчёта

Сохраним файл отчёта в нужном каталоге(см.рис.24)

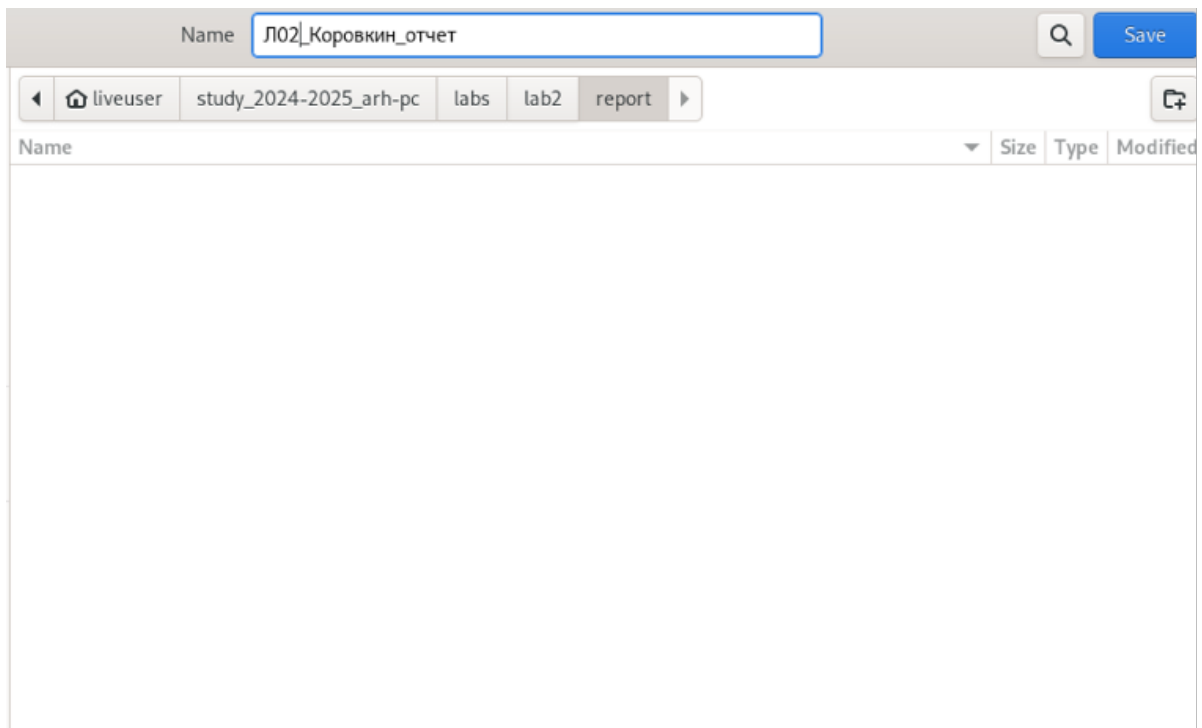


Рис. 3.2: Сохранение файла в нужном каталоге

Теперь, когда наш файл сохранен, скопируем отчёт о предыдущей лабораторной работе в каталог `labs/lab01/report`. Для этого воспользуемся командой `cp` (см.рис.25).

```
[liveuser@localhost-live Documents]$ cp Л01_Коровкин_отчет.pdf ~/home/study_2024-2025_arh-pc/labs/lab1/report
```

Рис. 3.3: Копируем файл в нужную директорию

Остаётся только отправить наши файлы и каталоги на сервер. Для этого сделаем те же самые действия, что и в основной части лабораторной работы. Сперва добавляем файлы и каталог, используем `commit` для утверждения изменений и отправляем на сервер при помощи `push` (см.рис.26-27).


```
[liveuser@localhost-live study_2024-2025_arh-pc]$ git add .
[liveuser@localhost-live study_2024-2025_arh-pc]$ git commit -am "feat(main):
added 2 labs"
[master cace897] feat(main): added 2 labs
 3 files changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 labs/lab1/report/Л01_Коровкин_отчет.pdf
 create mode 100644 labs/lab2/report/Л02_Коровкин_отчет.docx
 create mode 100644 labs/lab2/report/Л02_Коровкин_отчет.pdf
[liveuser@localhost-live study_2024-2025_arh-pc]$ git push
To github.com:Pancakeboy1987/study_2024-2025_arh-pc.git
```


Рис. 3.4: Использование команд add и commit


```
[liveuser@localhost-live study_2024-2025_arh-pc]$ git push
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (9/9), 9.85 KiB | 4.92 MiB/s, done.
Total 9 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 1 local object.
To github.com:Pancakeboy1987/study_2024-2025_arh-pc.git
   2a788b5..70ff901  master -> master
[liveuser@localhost-live study_2024-2025_arh-pc]$
```


Рис. 3.5: Загрузка файлов на сервер

Проверим, получилось ли правильно загрузить файлы на GitHub(см.рис.28). Как мы видим, нужные каталоги на месте. Работа выполнена верно.

study_2024-2025_arh-pc / labs / 

 Pancakeboy1987

Create read.me 

a9ab01c · 2 minutes ago  History




| Name | Last commit message | Last commit date |
|--|---------------------|------------------|
|  .. | | |
|  lab01 | Create read.me | 2 minutes ago |
|  lab02/report | feat(main): rename | 15 minutes ago |

Рис. 3.6: Загруженные каталоги с файлами

4 Выводы

В ходе выполнения данной лабораторной работы были получены навыки работы с системой контроля версий Git, такие как: первоначальная настройка системы, создание репозитория, изменение, сохранение и загрузка файлов на сервер.