

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

дисциплина: Архитектура компьютера

Студент: Попов Даниил Георгиевич

Группа: НПИбд-02-24

Студ. Билет №1132243109

МОСКВА

2024 г.

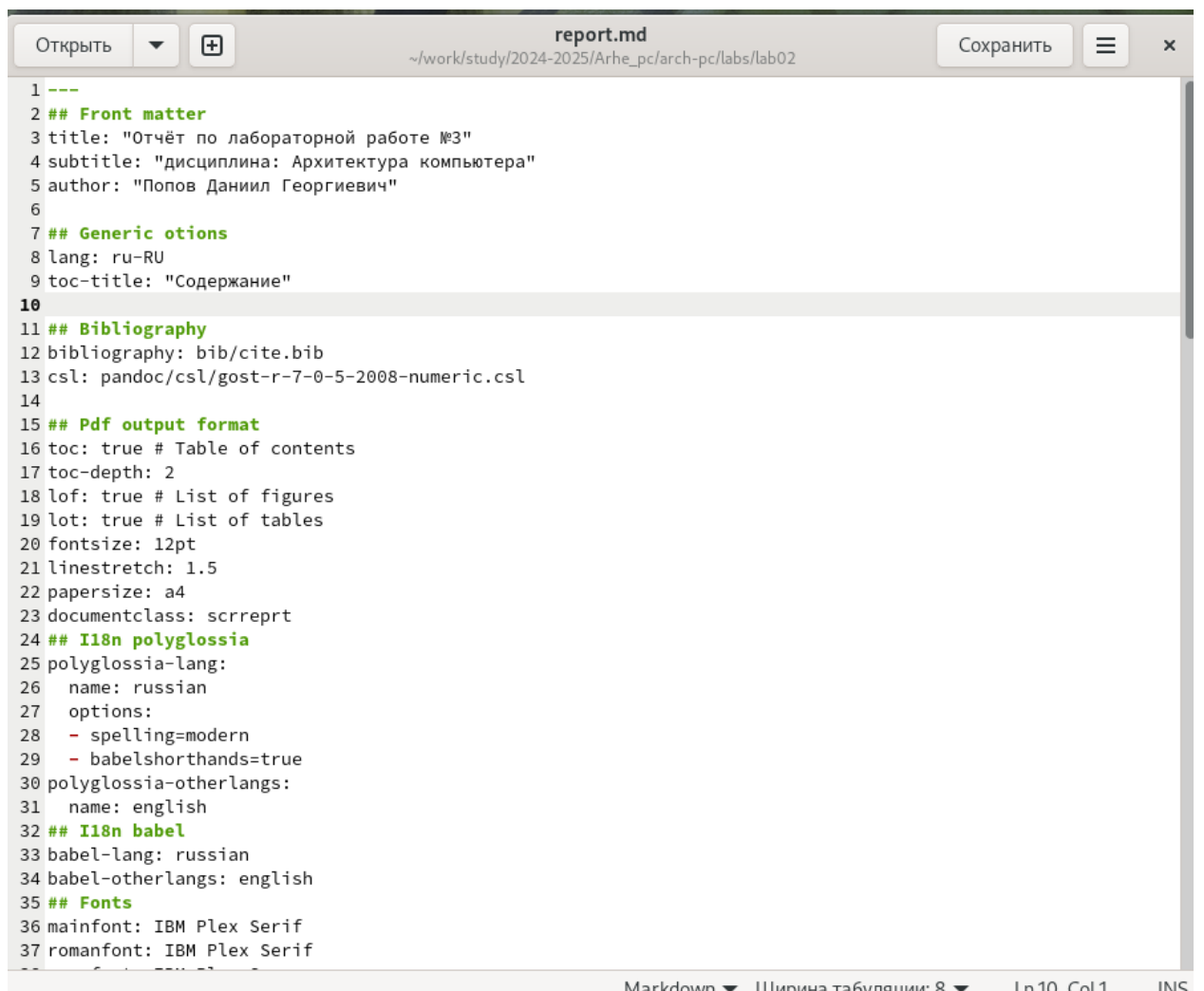
ВЫПОЛНЕНИЕ ЛАБОРАТОРНОЙ №3

Обновляем данные с помощью команды `git pull` и переходим в каталог с отчетом для создания файлов формата pdf и docx.

```
dpopov@fedora:~/work/study/2024-2025/Arhe_pc/arch-pc/labs/lab02$ cd
dpopov@fedora:~$ cd ~/work/study/2024-2025/Arhe_pc/arch-pc/labs/lab02
dpopov@fedora:~/work/study/2024-2025/Arhe_pc/arch-pc/labs/lab02$ make
pandoc "report.md" --filter pandoc/filters/pandoc_fignos.py --filter pandoc/filters/pandoc_eqnos.py --filter pandoc/filters/pandoc_tablenos.py --filter pandoc/filters/pandoc_secnos.py --number-sections --citeproc -o "report.docx"
pandoc "report.md" --filter pandoc/filters/pandoc_fignos.py --filter pandoc/filters/pandoc_eqnos.py --filter pandoc/filters/pandoc_tablenos.py --filter pandoc/filters/pandoc_secnos.py --pdf-engine=lualatex --pdf-engine-opt=--shell-escape --citeproc --number-sections -o "report.pdf"
```

Затем удаляем эти файлы с помощью команды `make clear`. Затем заходим в файл `report.md` и редактируем его.

```
dpopov@fedora:~/work/study/2024-2025/Arhe_pc/arch-pc/labs/lab02$ gedit report.md
```



ДЕЛАЕМ ОТЧЕТ ЛАБОРАТОРНОЙ №2

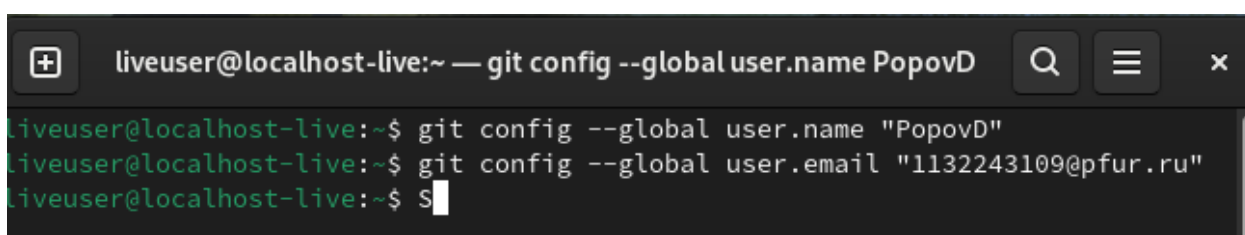
Цель работы

Ознакомится с системой контроля версий Git, настроить его, завести репозиторий на сайте github и скинуть в него свои отчеты по лабораторным работам.

Порядок выполнения работы:

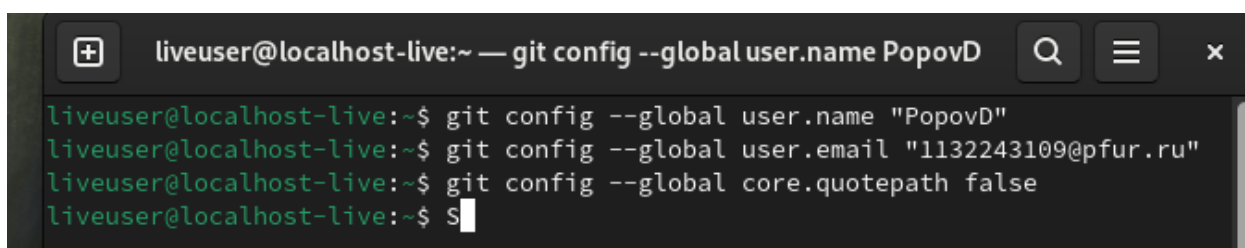
1. Базовая настройка Git:

Делаем предварительную конфигурацию git.



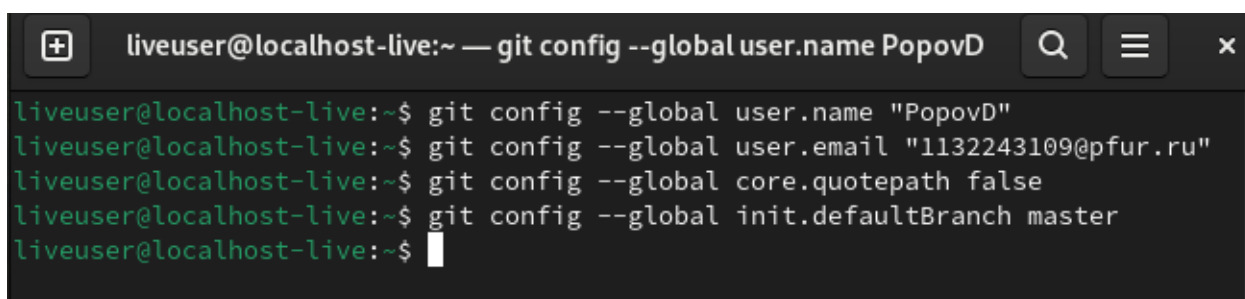
```
liveuser@localhost-live:~ — git config --global user.name PopovD
liveuser@localhost-live:~$ git config --global user.name "PopovD"
liveuser@localhost-live:~$ git config --global user.email "1132243109@pfur.ru"
liveuser@localhost-live:~$ S
```

Рис. 1.1 Задаем имя и email репозитория



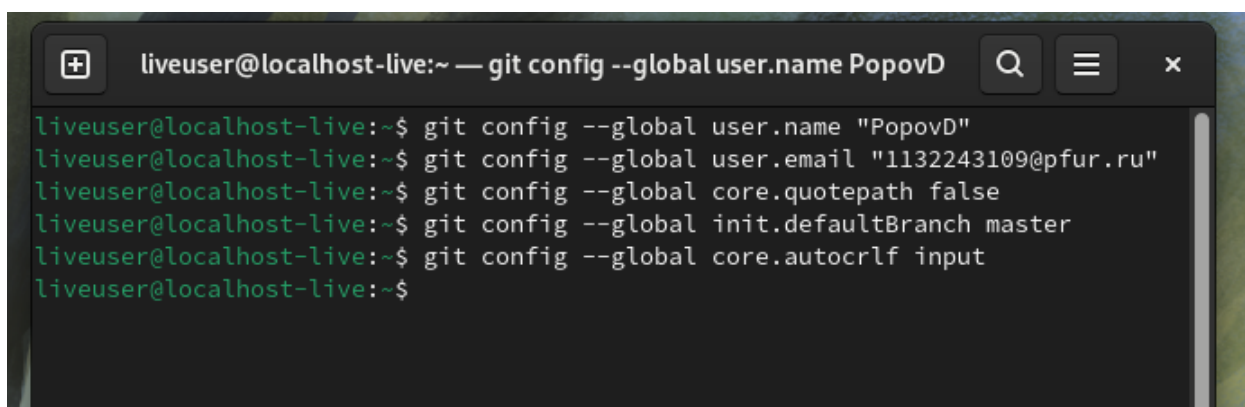
```
liveuser@localhost-live:~ — git config --global user.name PopovD
liveuser@localhost-live:~$ git config --global user.name "PopovD"
liveuser@localhost-live:~$ git config --global user.email "1132243109@pfur.ru"
liveuser@localhost-live:~$ git config --global core.quotepath false
liveuser@localhost-live:~$ S
```

Рис 1.2 Настраиваем utf-8



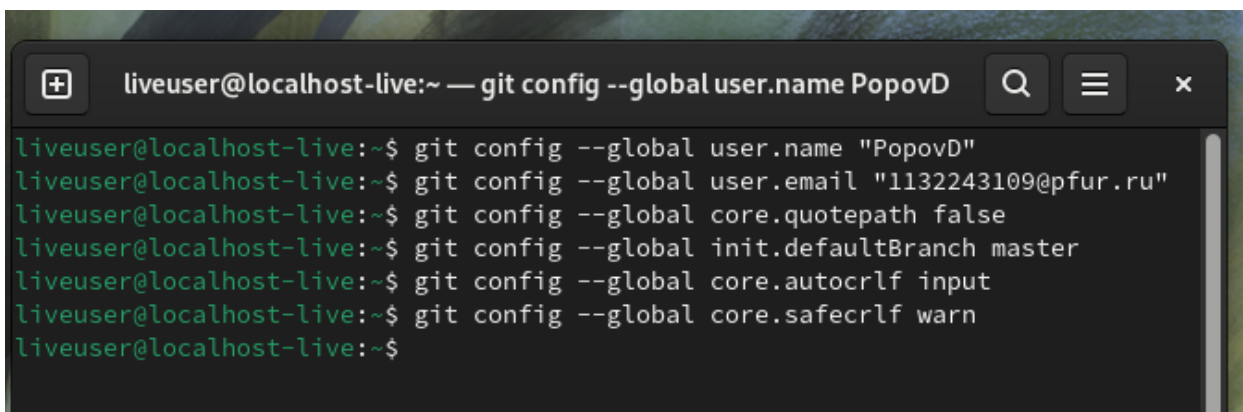
```
liveuser@localhost-live:~ — git config --global user.name PopovD
liveuser@localhost-live:~$ git config --global user.name "PopovD"
liveuser@localhost-live:~$ git config --global user.email "1132243109@pfur.ru"
liveuser@localhost-live:~$ git config --global core.quotepath false
liveuser@localhost-live:~$ git config --global init.defaultBranch master
liveuser@localhost-live:~$
```

Рис. 1.3 Задаем имя начальной ветки



```
liveuser@localhost-live:~ — git config --global user.name PopovD
liveuser@localhost-live:~$ git config --global user.name "PopovD"
liveuser@localhost-live:~$ git config --global user.email "1132243109@pfur.ru"
liveuser@localhost-live:~$ git config --global core.quotepath false
liveuser@localhost-live:~$ git config --global init.defaultBranch master
liveuser@localhost-live:~$ git config --global core.autocrlf input
liveuser@localhost-live:~$
```

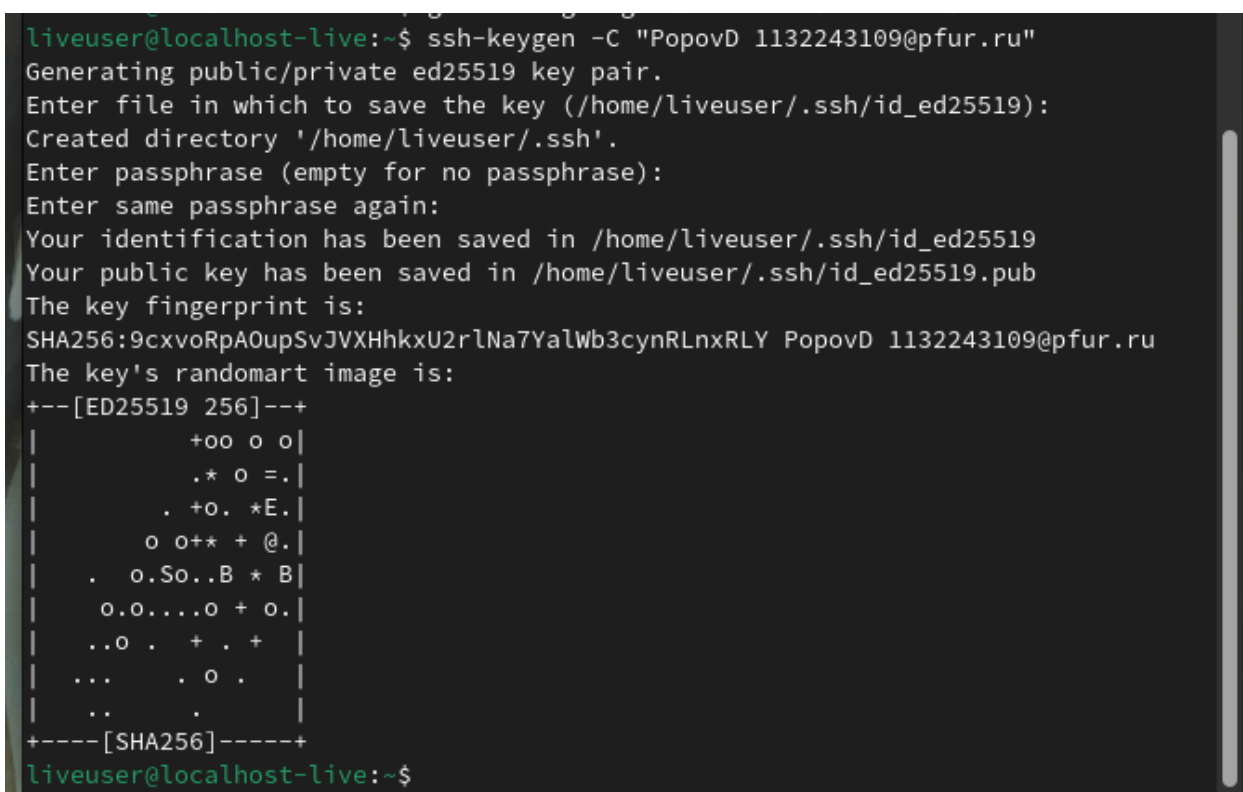
Рис. 1.4 Устанавливаем настройку autocrlf

A terminal window titled "liveuser@localhost-live:~ — git config --global user.name PopovD". It shows a series of git config commands being executed to set global configuration for a user named "PopovD".

```
liveuser@localhost-live:~$ git config --global user.name "PopovD"
liveuser@localhost-live:~$ git config --global user.email "1132243109@pfur.ru"
liveuser@localhost-live:~$ git config --global core.quotepath false
liveuser@localhost-live:~$ git config --global init.defaultBranch master
liveuser@localhost-live:~$ git config --global core.autocrlf input
liveuser@localhost-live:~$ git config --global core.safecrlf warn
liveuser@localhost-live:~$
```

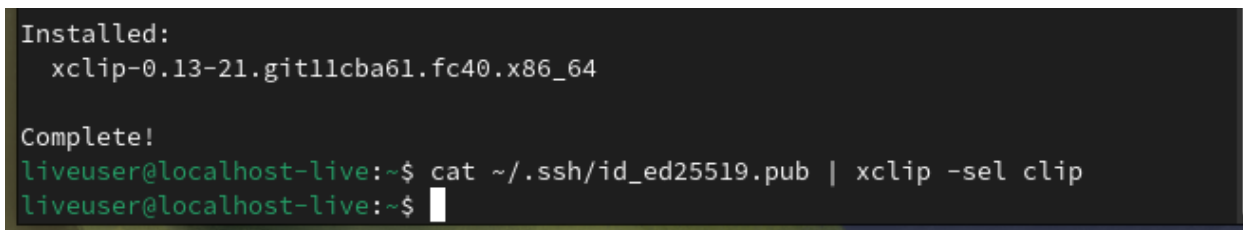
Рис. 1.5 Устанавливаем настройку safecrlf

2. Создание SSH ключа.

A terminal window showing the execution of the ssh-keygen command to generate an SSH key pair. The output includes prompts for filename, passphrase, and confirmation, followed by the key's fingerprint and a randomart image.

```
liveuser@localhost-live:~$ ssh-keygen -C "PopovD 1132243109@pfur.ru"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/liveuser/.ssh/id_ed25519):
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_ed25519
Your public key has been saved in /home/liveuser/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:9cxvoRpAOupSvJVXNhkxU2r1Na7YalWb3cynRLnxRLY PopovD 1132243109@pfur.ru
The key's randomart image is:
+--[ED25519 256]--+
|           +oo o o|
|           . * o =.|
|           . +o. *E.|
|           o o+* + @.|
|           . o.So..B * B|
|           o.o....o + o.|
|           ..o . + . +|
|           ... . o .|
|           ..      .|
+-----[SHA256]-----+
liveuser@localhost-live:~$
```

Рис. 2.1 Генерируем пару ключей

A terminal window showing the installation of xclip and the use of the cat command to copy the public key to the clipboard.

```
Installed:
  xclip-0.13-21.git11cba61.fc40.x86_64

Complete!
liveuser@localhost-live:~$ cat ~/.ssh/id_ed25519.pub | xclip -sel clip
liveuser@localhost-live:~$
```

Рис. 2.2 Скачиваем xclip и копируем в буфер обмена ключи

Заходим в свой аккаунт github и переходим в настройки:

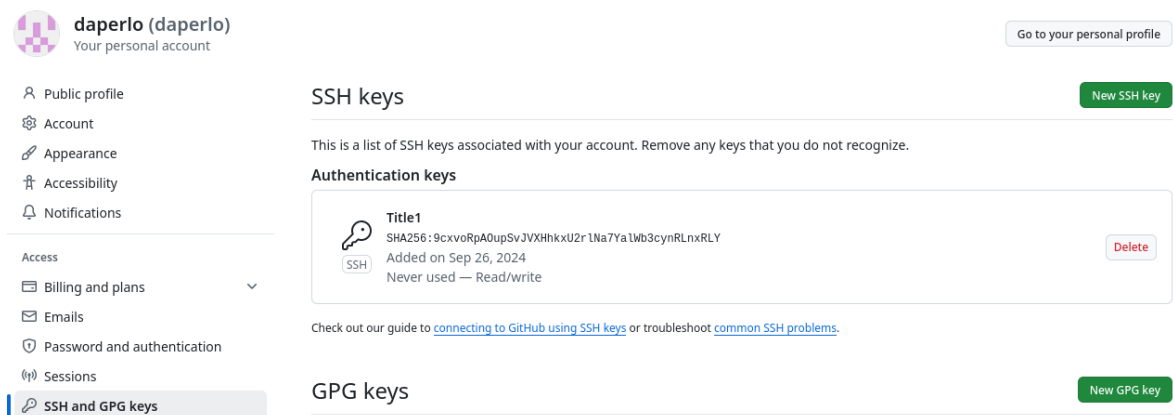


Рис. 2.3 Добавляем ключ и указываем имя ключа(Title1) и проверяем

3. Создание рабочего пространства и репозитория курса на основе шаблона.

```
liveuser@localhost-live:~$ mkdir -p ~/work/study/2024-2025/"Arhitectura pc"
liveuser@localhost-live:~$
```

Рис. 3.1 Создаем каталог для предмета “Архитектура компьютеров”


4. Создаем репозиторий курса.

Переходим на страницу репозитория с шаблоном

Create a new repository



A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk (*).

Owner *  daperlo / Repository name *
✔ study2024-2025_arh-pc is available.

Great repository names are short and memorable. Need inspiration? How about [ideal-happiness](#) ?

Description (optional)

- ☐  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☒  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

- ☐ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore

Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license

A license tells others what they can and can't do with your code. [Learn more about licenses](#).

Рис. 4.1 Создаем репозиторий по шаблону

```
liveuser@localhost-live:~$ cd ~/work/study/2024-2025/"Arhetictura pc"
liveuser@localhost-live:~/work/study/2024-2025/Arhetictura pc$ git clone --recur
sive git@
fatal: repository 'git@' does not exist
liveuser@localhost-live:~/work/study/2024-2025/Arhetictura pc$ git clone --recur
sive git@github.com:daperlo/study2024-2025_arh-pc.git arch-pc
Cloning into 'arch-pc'...
```

Рис. 4.2 Переходим в каталог и клонируем репозиторий

5. Настройка каталога курса

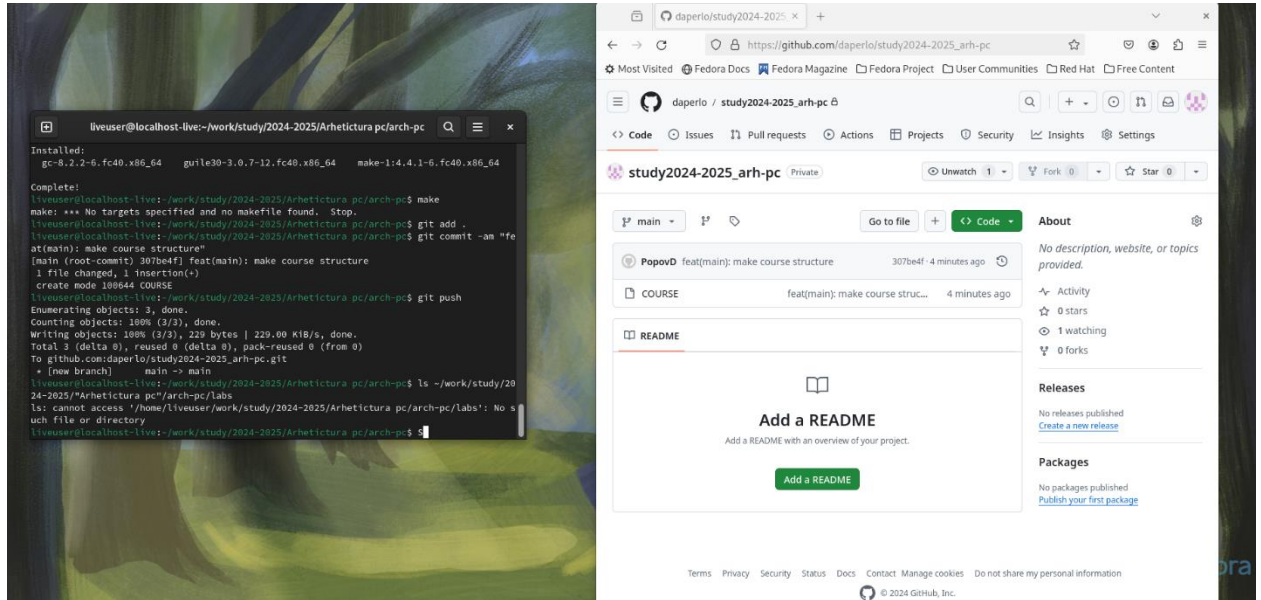
```
liveuser@localhost-live:~/work/study/2024-2025/Arhetictura pc$ git clone --recursive git@
github.com:daperlo/study2024-2025_arh-pc.git arch-pc
Cloning into 'arch-pc'...
warning: You appear to have cloned an empty repository.
liveuser@localhost-live:~/work/study/2024-2025/Arhetictura pc$ cd ~/work/study/2024-2025/
"Arhetictura pc"/arch-pc
liveuser@localhost-live:~/work/study/2024-2025/Arhetictura pc/arch-pc$ rm package.json
rm: cannot remove 'package.json': No such file or directory
liveuser@localhost-live:~/work/study/2024-2025/Arhetictura pc/arch-pc$ echo arch-pc > COU
RSE
```

Рис. 5.1 Переходим в каталог и создаем необходимые каталоги

Рис. 5.2 Отслеживаем файлы, отправляем в репозиторий и проверяем выполнение кода.

Вывод:

Мы познакомились с системой контроля git, выучили команды для работы с ним, создали свой репозиторий на платформе github, где в последующем будут



храниться все будущие отчеты по лабораторным работам.