

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

дисциплина: Архитектура компьютеров
и операционных систем

Студент: Дзаки Рафли
Группа: НБИбд-01–23

МОСКВА

2023 г

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	10
4.3	Создание SSH-ключа.....	11
4.4	Создание рабочего пространства и репозитория курса на основе шаблона	14
4.5	Создание репозитория курса на основе шаблона	15
4.6	Настройка каталога курса.....	17
4.7	Выполнение заданий для самостоятельной работы.....	20
5	Выводы	27
6	Список литературы	28

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 работ

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 заполняю основные данные учетной записи и регистрирую аккаунт.

Name

Your name may appear around GitHub where you contribute or are mentioned. You can remove it at any time.

Public email

You have set your email address to private. To toggle email privacy, go to [email settings](#) and uncheck "Keep my email address private."

Bio

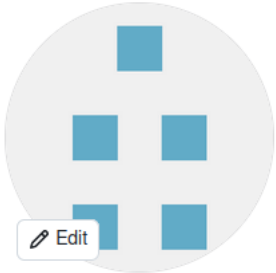
Profile picture


Рис. 4.1: Заполнение данных учетной записи GitHub

Аккаунт создан (рис. 4.2).

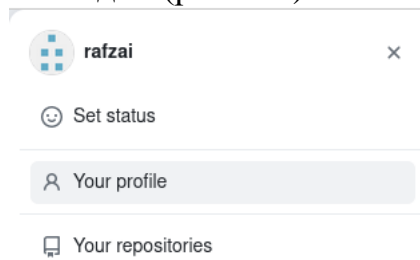


Рис. 4.2: Аккаунт GitHub

4.2 Базовая настройка Git

Запускаю виртуальную машину, затем в терминале задаю 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).

```
rzaidan@rzaidan-VirtualBox:~$ git config --global user.name rafzai
rzaidan@rzaidan-VirtualBox:~$ git config --global user.email 1032235550@pfur.ru
```

Рис. 4.3: Предварительная конфигурация git

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

```
rzaidan@rzaidan-VirtualBox:~$ git config --global core.quotepath false
```

Рис. 4.4: Настройка кодировки

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

```
rzaidan@rzaidan-VirtualBox:~$ git config --global init.defaultBranch master
```

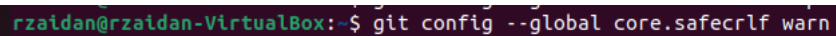
Рис. 4.5: Создание имени для начальной ветки

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.

```
rzaidan@rzaidan-VirtualBox:~$ 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.



```
rzaidan@rzaidan-VirtualBox: ~$ 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/`.

```
rzaidan@rzaidan-VirtualBox:~$ ssh-keygen -C "rafzai <1032235550@pfur.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/rzaidan/.ssh/id_rsa):
/home/rzaidan/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/rzaidan/.ssh/id_rsa
Your public key has been saved in /home/rzaidan/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:igmcONrXVwFCM3d6yZp6uZG7Vn8wATOLnzSPxAGI+SI rafzai <1032235550@pfur.ru>
The key's randomart image is:
+---[RSA 3072]-----+
|      +=.+.      |
|      o . = = .   |
|      . . o = *    |
| oE.. . . + * ..  |
|o +. . . S+ . = . |
|.o . . + o += +   |
|. . + + * . . o   |
| . . o . + . .   |
| . + . .         |
+-----[SHA256]-----+
```

Рис. 4.8: Генерация SSH-ключа

Xclip — утилита, позволяющая скопировать любой текст через терминал. Оказывается, в дистрибутиве Linux Ubuntu ее сначала надо установить. Устанавливаю xclip с помощью команды apt-get install с ключом -у от имени суперпользователя, введя в начале команды sudo (рис. 4.9).

```
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/computer architecture$ git clone --recursive git@github.com:rafzai/study_2023-2024_arch-pc.git
Cloning into 'study_2023-2024_arch-pc'...
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpZisF/zLDA0zPMSvHdKr4UvCOqU.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
git@github.com: Permission denied (publickey).
fatal: Could not read from remote repository.
```

Рис. 4.9: Установка утилиты xclip

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

```
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/computer architecture$ ssh-keygen -C "rafzai <1032235550@pfur.ru>"
```

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.

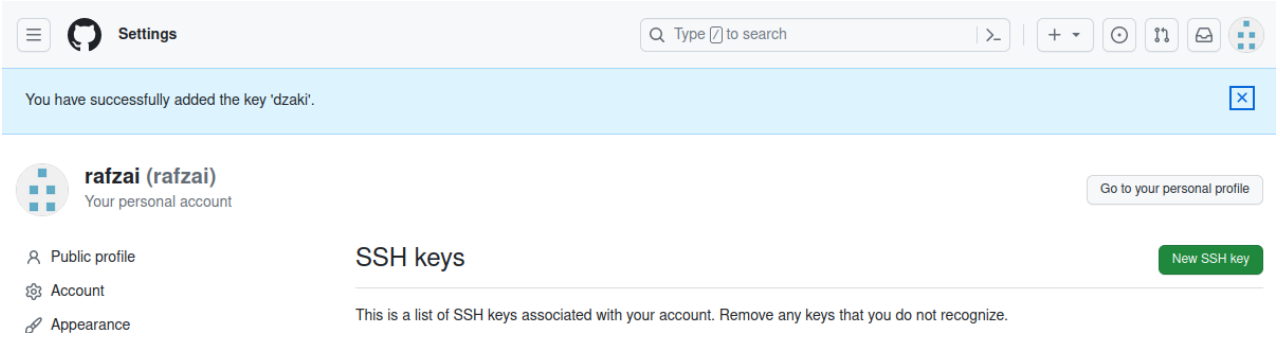


Рис. 4.11: Окно SSH and GPG keys

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

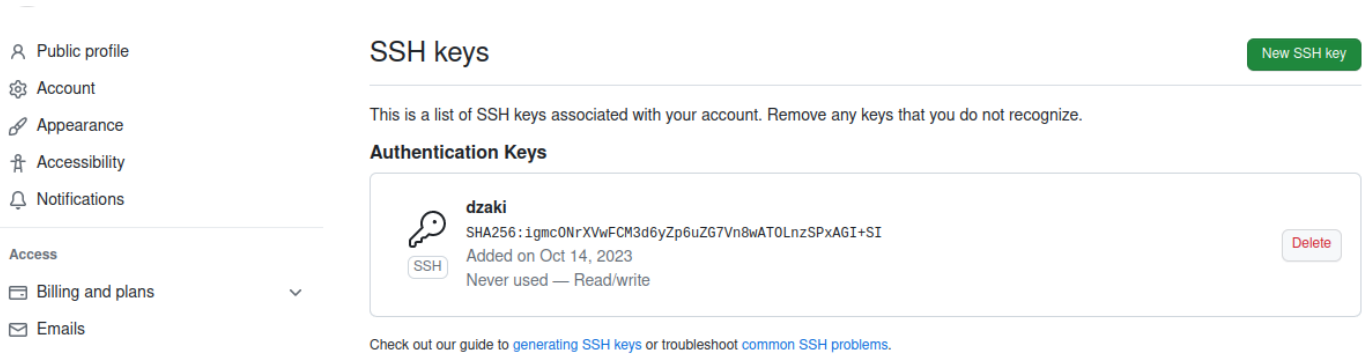


Рис. 4.12: Добавление ключа

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/"Computer architecture"` рекурсивно. Далее проверяю с помощью `ls`, действительно ли были созданы необходимые мне каталоги (рис. 4.13).

```
rzaidan@rzaidan-VirtualBox:~$ mkdir -p ~/work/study/2023-2024/"Computer architecture"
rzaidan@rzaidan-VirtualBox:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  snap  Templates  Videos  work
```

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/yamadharm/course-directory-student-template>.

Далее выбираю «Use this template», чтобы использовать этот шаблон для своего репозитория (рис.4.14).

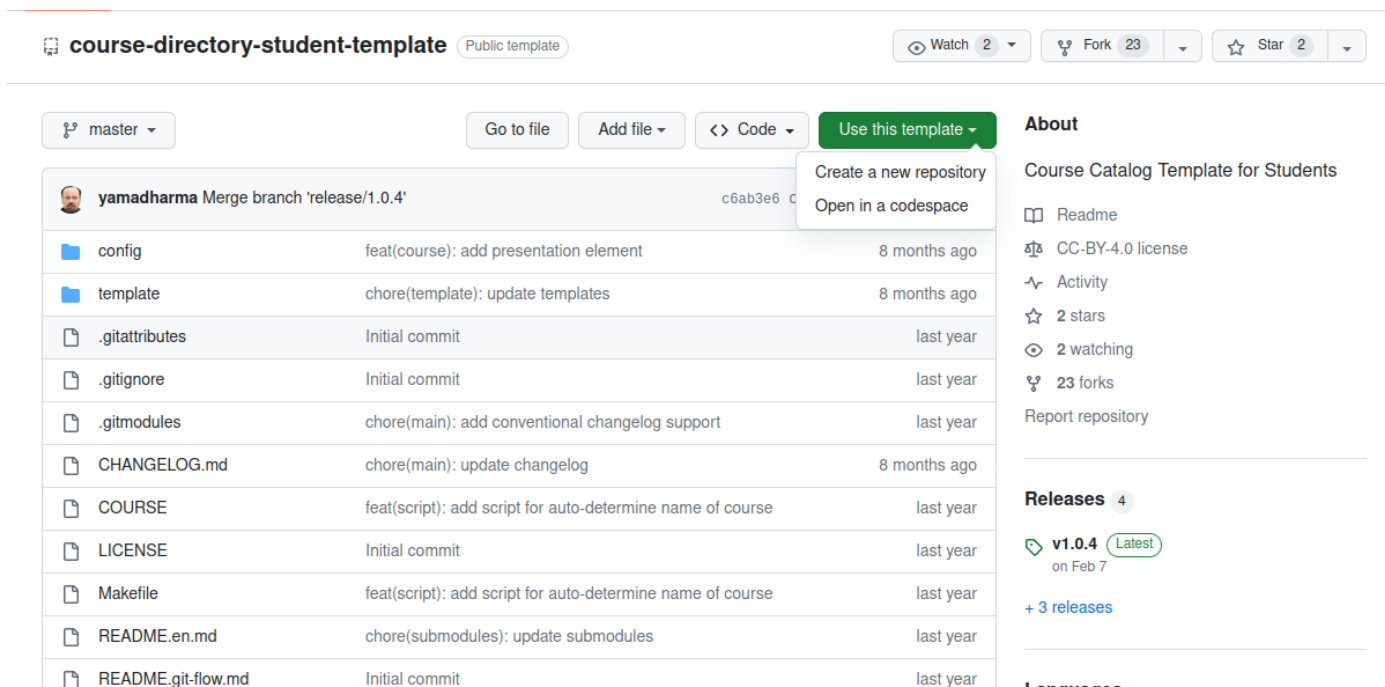


Рис. 4.14: Страница шаблона для репозитория

В открывшемся окне задаю имя репозитория (Repository name): study_2022–2023_arch-рс и создаю репозиторий, нажимаю на кнопку «Create repository» (рис. 4.15).

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 * takmuradow / **Repository name *** study_2023-2024_arch-pc

✔ study_2023-2024_arch-pc is available.

Great repository names are short and memorable. Need inspiration? How about **glowing-bassoon** ?

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.

☐ You are creating a public repository in your personal account.

Create repository

Рис. 4.15: Окно создания репозитория

master
 1 branch
 0 tags

Go to file
 Add file
 Code

rafzai Add files via upload
 2f654c8 35 minutes ago 10 commits

config	Initial commit	8 hours ago
labs	Add files via upload	35 minutes ago
presentation	feat(main): make course structure	3 hours ago
template	Initial commit	8 hours ago
.gitattributes	Initial commit	8 hours ago
.gitignore	Initial commit	8 hours ago
.gitmodules	Initial commit	8 hours ago
CHANGELOG.md	Initial commit	8 hours ago
COURSE	feat(main): make course structure	3 hours ago
LICENSE	Initial commit	8 hours ago
Makefile	Initial commit	8 hours ago
README.en.md	Initial commit	8 hours ago
README.git-flow.md	Initial commit	8 hours ago

About

No description, website, or topics provided.

- Readme
- CC-BY-4.0 license
- Activity
- 0 stars
- 1 watching
- 0 forks

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

Languages

Рис. 4.16: Созданный репозиторий

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

```

rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture$ cd ~/work/study/2023-2024/"Computer architecture"/arch-pc
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture/arch-pc$
  
```

Рис. 4.17: Перемещение между директориями

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH» (рис. 4.18).

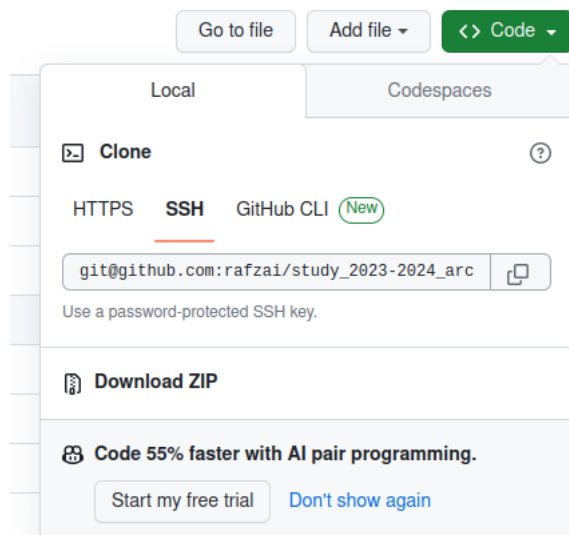


Рис. 4.18: Окно с ссылкой для копирования репозитория

Клонирую созданный репозиторий с помощью команды

```
git clone --recursive git@github.com:/study_2023-2024_arh-pc.git arch-pc
```

(рис. 4.19).

```
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture$ git clone --recursive git@github.com:rafzai/study_2023-2024
_arch-pc.git arch-pc
Cloning into 'arch-pc'...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 27 (delta 1), reused 11 (delta 0), pack-reused 0
Receiving objects: 100% (27/27), 16.93 KiB | 234.00 KiB/s, done.
Resolving deltas: 100% (1/1), done.
Submodule 'template/presentation' (https://github.com/yamadharma/academic-presentation-markdown-template.git) registered for path 't
emplate/presentation'
Submodule 'template/report' (https://github.com/yamadharma/academic-laboratory-report-template.git) registered for path 'template/re
port'
Cloning into '/home/rzaidan/work/study/2023-2024/Computer architecture/arch-pc/template/presentation'...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Receiving objects: 100% (82/82), 92.90 KiB | 44.00 KiB/s, done.
Resolving deltas: 100% (28/28), done.
Cloning into '/home/rzaidan/work/study/2023-2024/Computer architecture/arch-pc/template/report'...
remote: Enumerating objects: 101, done.
remote: Counting objects: 100% (101/101), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 101 (delta 40), reused 88 (delta 27), pack-reused 0
Receiving objects: 100% (101/101), 327.25 KiB | 109.00 KiB/s, done.
Resolving deltas: 100% (40/40), done.
Submodule path 'template/presentation': checked out 'b1be3800ee91f5809264cb755d316174540b753e'
Submodule path 'template/report': checked out '1d1b61dcac9c287a83917b82e3aef11a33b1e3b2'
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture$
```

Рис. 4.19: Клонирование репозитория

4.6 Настройка каталога курса

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

```
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture$ cd ~/work/study/2023-2024/"Computer architecture"/arch-pc
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture/arch-pc$
```

Рис. 4.20: Перемещение между директориями

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

```
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture/arch-pc$ rm package.json
```

Рис. 4.21: Удаление файлов

Создаю необходимые каталоги (рис. 4.22).

```
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture/arch-pc$ echo arch-pc > COURSE
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture/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).

```
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture/arch-pc$ git add .
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture/arch-pc$ git commit -am 'feat(main): make course structure'
[master 0c2dbb6] 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
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/core.py
```

Рис. 4.23: Добавление и сохранение изменений на сервере

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

```
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture/arch-pc$ git push
Enumerating objects: 37, done.
Counting objects: 100% (37/37), done.
Delta compression using up to 8 threads
Compressing objects: 100% (29/29), done.
Writing objects: 100% (35/35), 342.13 KiB | 1.07 MiB/s, done.
Total 35 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:rafzai/study_2023-2024_arch-pc.git
873ec6c..0c2dbb6 master -> master
```

Рис. 4.24: Выгрузка изменений на сервер

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

The screenshot displays the GitHub interface for the repository 'study_2023-2024_arch-pc' by user 'rafzai'. The left sidebar shows the file tree with the 'labs' directory selected. The main area shows the commit history for the 'labs' directory, listing folders 'lab01' through 'lab09' (partially visible). Each folder has a commit message 'feat(main): make course structure' and a commit date of '4 minutes ago'.

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

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 третьей лабораторной работе с помощью утилиты touch (рис. 4.26).

```
rzaidan@rzaidan-VirtualBox:~/work/study/2023-2024/Computer architecture/arch-pc/labs/lab02/report$ touch L02_rafbai_Report
```

Рис. 4.26: Создание файла

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

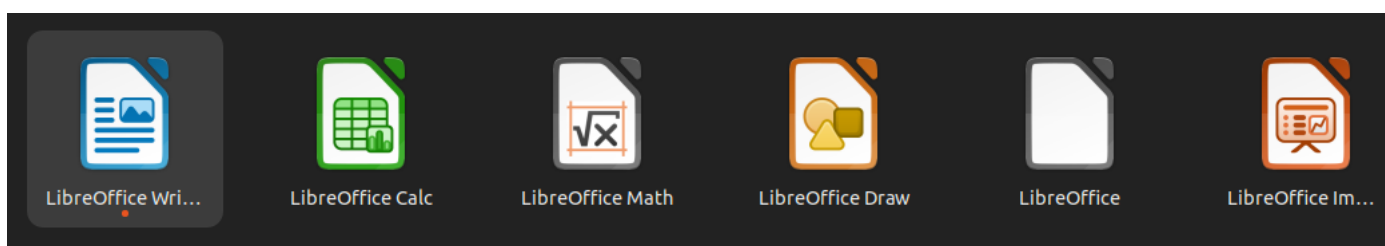


Рис. 4.27: Меню приложений

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

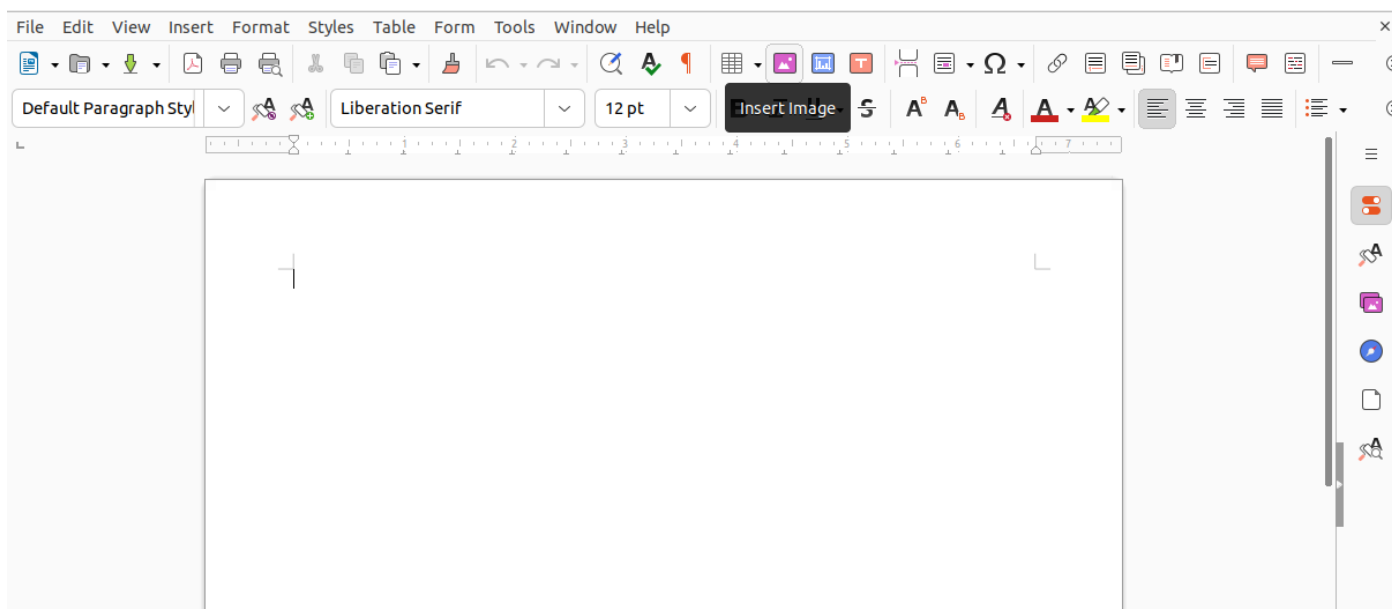


Рис. 4.28: Работа с отчетом в текстовом редакторе

5 Выводы

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

6 Список литературы

1. Архитектура ЭВМ
2. Git - gitattributes Документация

