

# **ОТЧЕТ**

## **ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2**

Дисциплина: Архитектура компьютера

**Луангсуваннавонг Сайпхачан**

Студентческий билет: 1032249112

Группа: НКАбд 01-24

# Содержание

1 Цель работы	3
2 Задание	4
3 Теоретическое введение	5
4 Выполнение лабораторной работы	7
4.1 Настройка GitHub	7
4.2 Базовая настройка Git	7
4.3 Создание SSH ключа	8
4.4 Создание рабочего пространства и репозитория курса на основе шаблона	11
4.5 Создание репозитория курса на основе шаблона	11
4.6 Настройка каталога курса	13
5 Выполнение заданий для самостоятельной работы	15
6 Выводы	20
7 Источники	21

# 1. Цель работы

Целью данной работы является изучение и понимание применения инструментов контроля версий, идеологии приложения, а также получение практических навыков работы 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 (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.

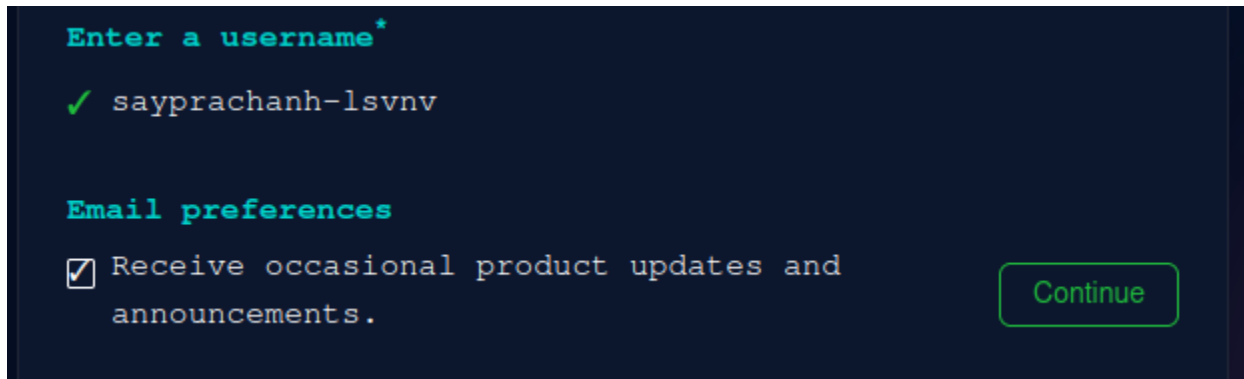
Системы контроля версий также могут обеспечивать дополнительные, более гибкие функциональные возможности. Например, они могут поддерживать работу с несколькими версиями одного файла, сохраняя общую историю изменений до точки ветвления версий и собственные истории изменений каждой ветви. Кроме того, обычно 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 Git представляет собой набор командных инструментов, к которым можно получить доступ через терминал, вводя команду git с различными опциями. Поскольку Git является распределённой системой контроля версий, создать резервную копию локального репозитория можно простым копированием или архивированием. Работа

пользователя с его веткой начинается с проверки и получения изменений из центрального репозитория, при этом перед этой процедурой в локальное дерево не должно быть внесено никаких изменений. После этого пользователи могут вносить изменения в своё локальное дерево и/или ветку. Когда изменения в файлах и/или каталогах проекта завершены, их необходимо загрузить в центральный репозиторий.

## 4. Выполнение лабораторной работы

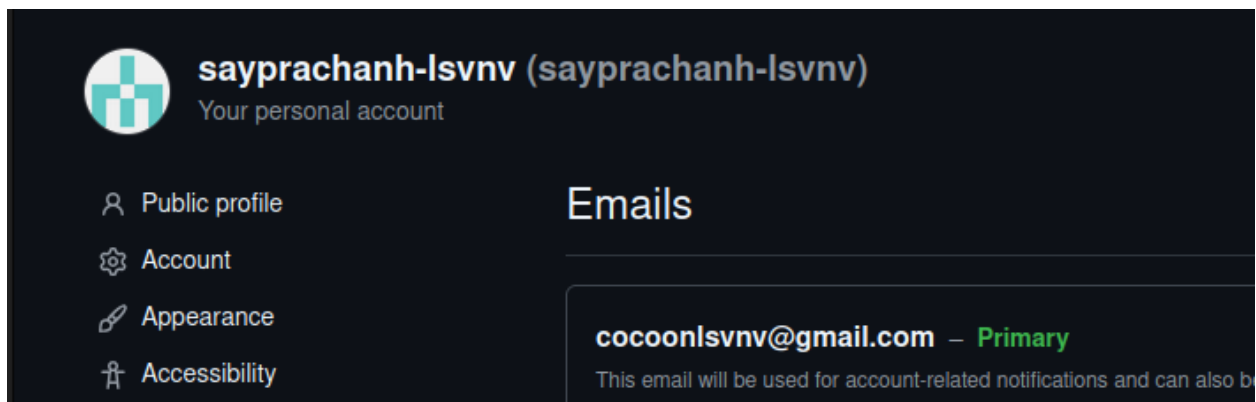
### 4.1 Настройка GitHub

Я создаю учетную запись на сайте GitHub, а также заполнил основные реквизиты учетной записи. (Рис .4.1)

A screenshot of the GitHub account creation interface. It features a dark blue background with light blue and white text. The first section is titled "Enter a username\*" and shows a green checkmark next to the username "sayprachanh-lsvnv". The second section is titled "Email preferences" and includes a checked checkbox for "Receive occasional product updates and announcements.". A green "Continue" button is located on the right side of the form.

(Рис .4.1 Заполнение данных учетной записи на GitHub)

Аккаунт создан (Рис .4.2)



(Рис .4.2 Аккаунт GitHub)

### 4.2 Базовая настройка Git

Открываю терминал и произвожу предварительную настройку git. Я ввожу команду "git config --global user.name", указывая свое имя, и команду "git config --global user.email", указывая в ней свой адрес электронной почты. (Рис .4.3)

```
[sayprachanh@archlinux ~]$ git config --global user.name "<Sayprachanh Luangsouvannavong>"
[sayprachanh@archlinux ~]$ git config --global user.email "<cocoonslvnv@gmail.com>"
[sayprachanh@archlinux ~]$
```

(Рис .4.3 Предварительная настройка Git)

Я настраиваю utf-8 в выходных данных сообщений git для корректного отображения имен файлов и символов (Рис .4.4)

```
[sayprachanh@archlinux ~]$ git config --global core.quotepath false
[sayprachanh@archlinux ~]$
```

(Рис .4.4 Установка кодировки)

имя для начальной ветки (в этом случае я называю ее "master"). (Рис .4.5)

```
[sayprachanh@archlinux ~]$ git config --global init.defaultBranch master
[sayprachanh@archlinux ~]$
```

(Рис .4.5 Установка имени для начальной ветки)

Я задаю параметру autocrlf со значением input (Рис .4.6), поскольку я работаю в системе Linux, я гарантирую, что файлы будут Конвертировать из CRLF в LF при коммитах. CR и LF - это символы, которые используются для обозначения разрыва строки в тексте/файлах.

```
[sayprachanh@archlinux ~]$ git config --global core.autocrlf input
[sayprachanh@archlinux ~]$
```

(Рис .4.6 Параметр autocrlf)

Используя команду "git config --global core.safecrlf" и значение "warn", я настраиваю Git так, чтобы он выдавал предупреждение, когда я пытаюсь получить доступ к файлам за пределами текущего репозитория. (Рис .4.7)

```
[sayprachanh@archlinux ~]$ git config --global core.safecrlf warn
[sayprachanh@archlinux ~]$
```

(Рис .4.7 Параметр safecrlf)

## 4.3 Создание SSH ключа

Потом, для последующей идентификации пользователя на сервере репозитория необходимо сгенерировать пару ключей (приватный и открытый). Для этого я использую команду "ssh-keygen -C "имя, фамилия", work@email", а также указываю имя и электронную почту владельца. Ключ будет автоматически сохранен в каталоге "~/.ssh". (Рис 4.8)



```
sayprachanh@archlinux:~$ ssh-keygen -C "Sayprachanh Luangsouvannavong <cocoonlsvnv@gmail.com>"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/sayprachanh/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/sayprachanh/.ssh/id_ed25519
Your public key has been saved in /home/sayprachanh/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:RHmDnBDu4nF0o1QWiBMBfmPRr9GkUU4JSsxmKI40/PY Sayprachanh Luangsouvannavong <cocoonlsvnv@gmail.com>
The key's randomart image is:
+--[ED25519 256]--+
| . .==+BB*      |
| ..B+++B o      |
| +.+O=.*+=. .   |
| .. =o+o.        |
| . + +oS         |
| . E.            |
| .               |
+-----[SHA256]-----+
[sayprachanh@archlinux ~]$
```

(Рис .4.8 Генерация SSH-ключа)

Xclip - это утилита, которая позволяет копировать текст из терминала.

Но, к сожалению, в дистрибутиве Arch Linux ее необходимо устанавливать отдельно. Я использую команду `pacman -S` для установки Xclip и ввожу `sudo` в начале команды от имени суперпользователя (Рис .4.9)

```
[sayprachanh@archlinux ~]$ sudo pacman -S xclip
[sudo] password for sayprachanh:
resolving dependencies...
looking for conflicting packages...

Packages (1) xclip-0.13-5

Total Download Size: 0.01 MiB
Total Installed Size: 0.03 MiB

:: Proceed with installation? [Y/n] y
:: Retrieving packages...
xclip-0.13-5-x86_64 14.7 KiB 130 KiB/s 00:00 [#####] 100%
(1/1) checking keys in keyring [#####] 100%
(1/1) checking package integrity [#####] 100%
(1/1) loading package files [#####] 100%
(1/1) checking for file conflicts [#####] 100%
(1/1) checking available disk space [#####] 100%
:: Processing package changes...
(1/1) installing xclip [#####] 100%
:: Running post-transaction hooks...
(1/1) Arming ConditionNeedsUpdate...
[sayprachanh@archlinux ~]$
```

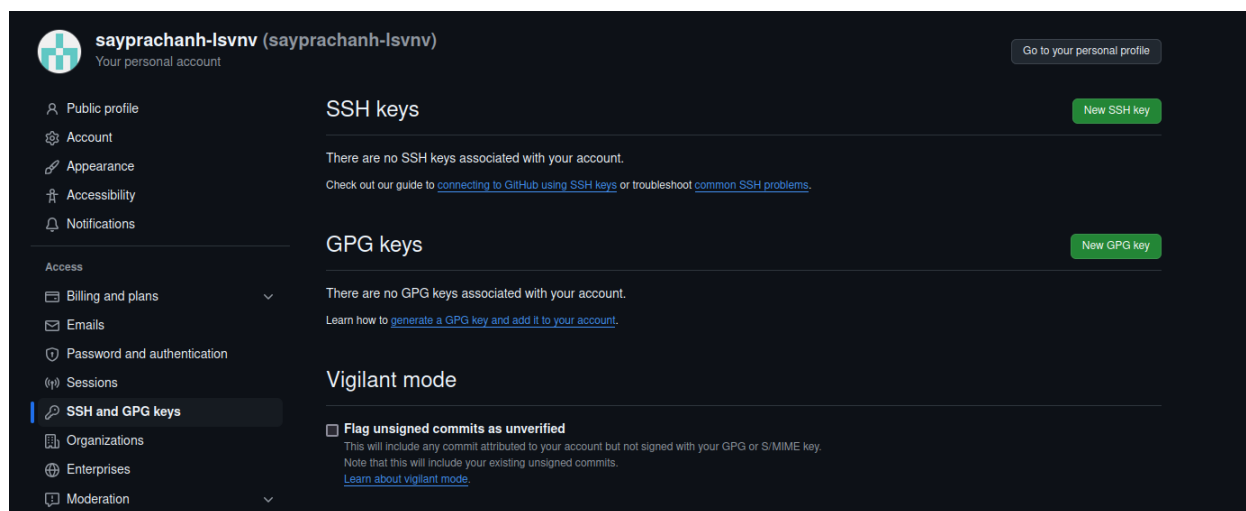
(Рис .4.9 Установка Xclip)

После установки Xclip, я продолжаю лабораторную работу, используя команды `cat` и `xclip`, я копирую ключ из каталога `"/.ssh/id_ed25519.pub"`, в котором он был сохранен. (Рис .4.10)

```
[sayprachanh@archlinux ~]$ cat ~/.ssh/id_ed25519.pub | xclip -sel clip
[sayprachanh@archlinux ~]$
```

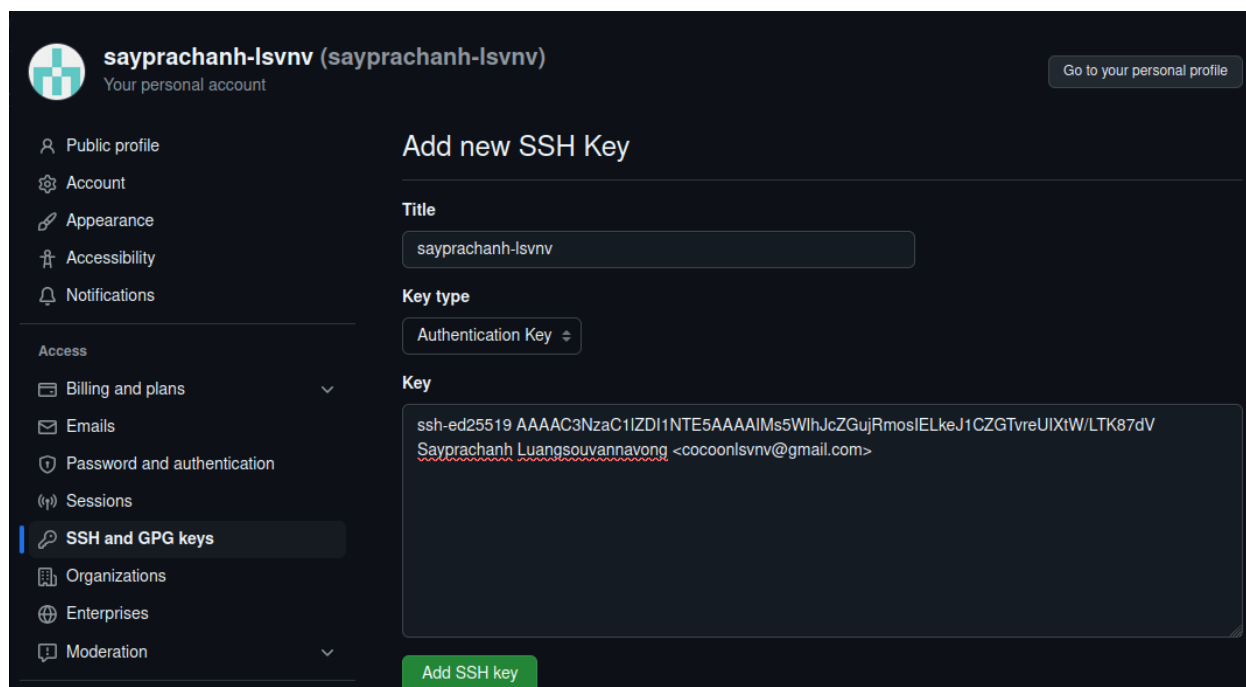
(Рис .4.10 Копирование содержимого файла)

Я открываю браузер и захожу на сайт GitHub, затем захожу в свой профиль и выбираю страницу "SSH и GPG Keys", нажимаю кнопку "New SSH key", чтобы добавить SSH ключ, который копирую с терминала (Рис .4.11)



(Рис .4.11 Страница SSH and GPG keys)

Вставляю ключ в раздел Key, а также задаю название SSH-ключа, затем нажимаю "Add SSH key", чтобы завершить добавление ключа (Рис .4.12)



(Рис .4.12 Добавление SSH-ключа)

## 4.4 Сознание рабочего пространства и репозитория курса на основе шаблона

После этого, используя утилиту `mkdir` и опцию `-p`, я создаю каталог рабочего пространства, после `home` `"~/work/study/2024-2025/Архитектура компьютера"`, который я создаю рекурсивно. Затем, используя команду `ls`, я проверяю правильность выполнения команды (Рис .4.13)

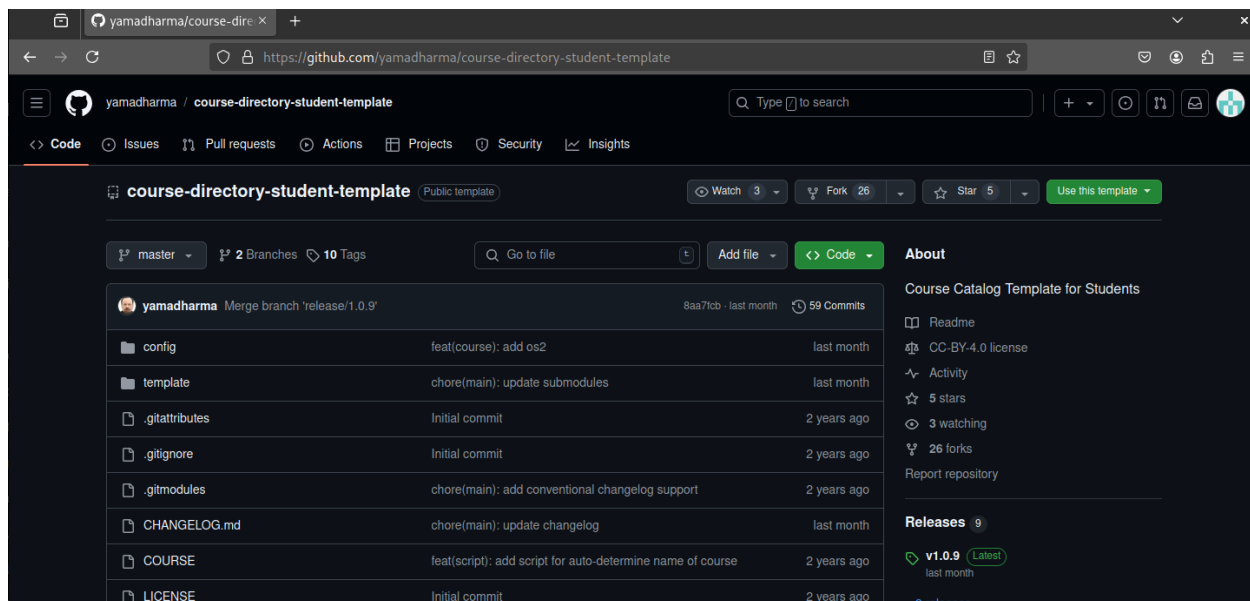
```
[sayprachanh@archlinux ~]$ mkdir -p work/study/2024-2025/"Архитектура компьютера"
[sayprachanh@archlinux ~]$ ls
Desktop Documents Downloads Music newdir Pictures Public temp Templates tmp Videos work
```

(Рис .4.13 Создание каталога для работы)

## 4.5 Сознание репозитория курса на основе шаблона

Я перехожу на страницу репозитория с шаблоном курса `"https://github.com/yamadharma/course-directory-student-template"` в браузере.

Затем я выбираю `"Use this template"`, чтобы использовать этот шаблон для своего репозитория (Рис .4.14)



(Рис .4.14 Страница шаблона для репозитория)

В открывшемся окне, я задаю имя репозитория (repository name) `"study_2024-2025_arh-rc"` и создаю репозиторий, нажав на кнопку `"Create repository"` (Рис .4.15)

Owner \* sayprachanh-lsvnv / Repository name \* study\_2024-2025\_arh-pc

✔ study\_2024-2025\_arh-pc is available.

Great repository names are short and memorable. Need inspiration? How about **bookish-adventure** ?

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.

*(i)* You are creating a public repository in your personal account.

[Create repository](#)

(Рис .4.15 создание репозитория)

Репозиторий создан (Рис .4.16)

master Go to file + <> Code

**sayprachanh-lsvnv** Initial commit 479fa13 · now 🕒 1 Commit

📁 config	Initial commit	now
📁 template	Initial commit	now
📄 .gitattributes	Initial commit	now
📄 .gitignore	Initial commit	now
📄 .gitmodules	Initial commit	now
📄 CHANGELOG.md	Initial commit	now
📄 COURSE	Initial commit	now
📄 LICENSE	Initial commit	now

(Рис .4.16 Созданный репозиторий)

Затем я захожу в терминал, используя утилиту `cd`, перехожу в каталог `"~/work/study/2024-2025/Архитектура компьютера"`. (Рис .4.17)

```
sayprachanh@archlinux:~/work/study/2024-2025/Архитектура компьютера
[sayprachanh@archlinux ~]$ cd ~/work/study/2024-2025/"Архитектура компьютера"
[sayprachanh@archlinux Архитектура компьютера]$
```

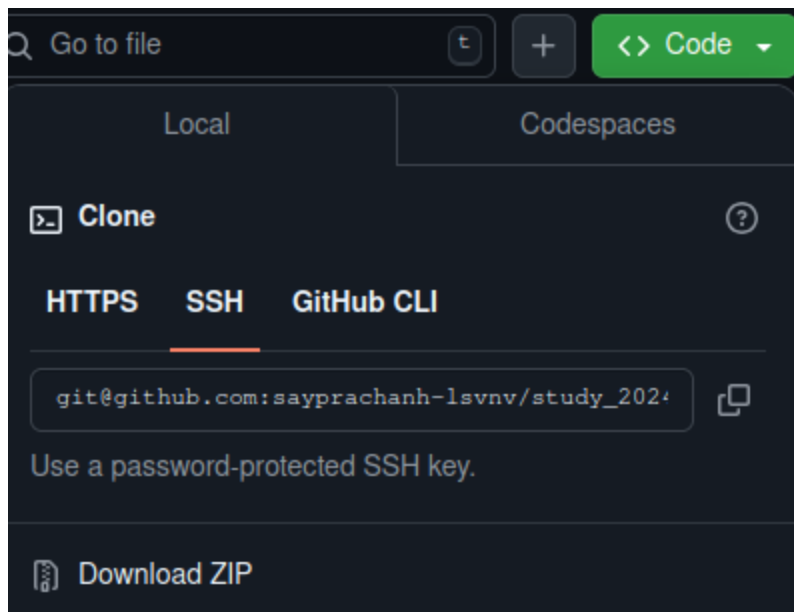
(Рис .4.17 Перемещение между каталогами)

Я клонирую созданный репозиторий, используя команду `"git clone -recursive git@github.com:sayprachanh-lsvnv/study_2024-2025_arh-pc.git arch-pc"`. (Рис .4.18)

```
[sayprachanh@archlinux Архитектура компьютера]$ git clone --recursive git@github.com:sayprachanh-lsvnv/study_2024-2025_arh-pc.git arch-pc
Cloning into 'arch-pc'...
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
```

(Рис .4.18 Клонирование репозитория)

Я копирую ссылку для клонирования на страницу созданного репозитория, перейдя в окно `"Code"`, затем выбираю `"SSH"` (Рис .4.19)



(Рис .4.19 Окно со ссылкой для клонирования репозитория)

## 4.6 Настройка каталога курса

Я перехожу в каталог `arch-pc` с помощью утилиты `cd` (Рис .4.20)

```
[sayprachanh@archlinux Архитектура компьютера]$ cd ~/work/study/2024-2025/'Архитектура компьютера'/arch-pc
[sayprachanh@archlinux arch-pc]$
```

(Рис .4.20 Перемещение между каталогами)

команду `rm`, я удаляю ненужные файлы, а именно файл "package.json". (Рис .4.21)

```
[sayprachanh@archlinux arch-pc]$ rm package.json
[sayprachanh@archlinux arch-pc]$
```

(Рис .4.21 Удаление файла)

Создайте необходимые каталоги (Рис .4.22)

```
[sayprachanh@archlinux arch-pc]$ echo arch-pc > COURSE
[sayprachanh@archlinux arch-pc]$ make
```

(Рис .4.22 Создание каталогов)

Затем я отправляю локальный каталог и файлы на сервер: используя команду "git add .", я добавляю все созданные каталоги. Затем, используя `git commit`, я сохраняю изменения и комментирую их на сервере. (Рис .4.23)

```
[sayprachanh@archlinux arch-pc]$ git add .
[sayprachanh@archlinux arch-pc]$ git commit -am 'feat(main): make course structure'
[master 9f1c2b4] feat(main): make course structure
223 files changed, 53681 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/.projectile
create mode 100644 labs/lab01/presentation/.texlabroot
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/.projectile
create mode 100644 labs/lab02/presentation/.texlabroot
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
```

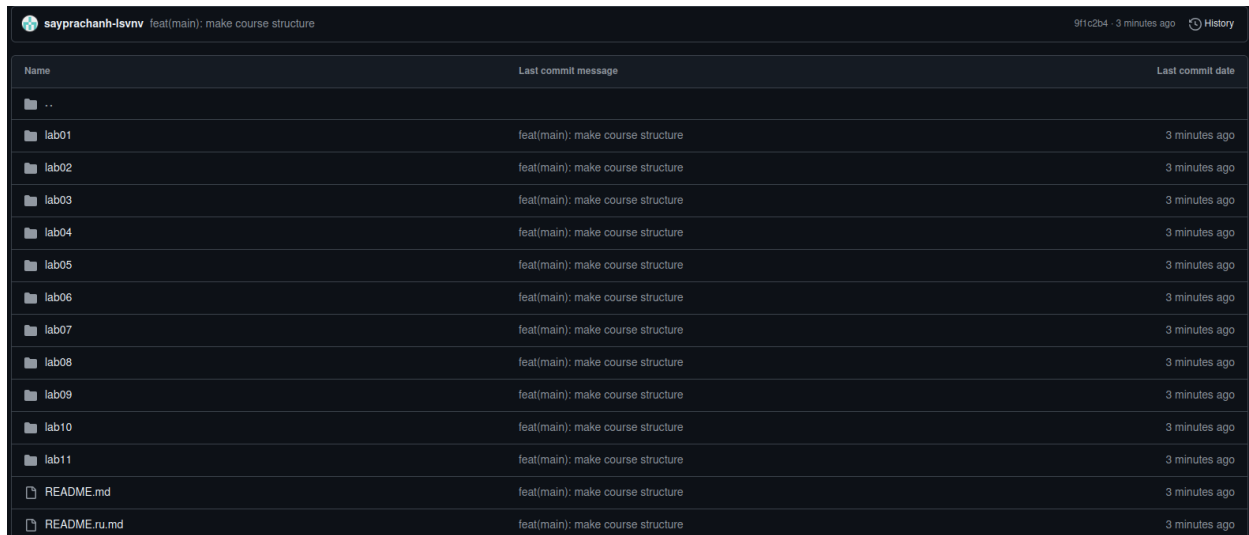
(Рис .4.23 Добавление и сохранение изменений на сервере)

отправьте все это на сервер с помощью команды `git push` (Рис .4.24)

```
[sayprachanh@archlinux arch-pc]$ git push
Enumerating objects: 37, done.
Counting objects: 100% (37/37), done.
Delta compression using up to 4 threads
Compressing objects: 100% (29/29), done.
Writing objects: 100% (35/35), 341.29 KiB | 2.37 MiB/s, done.
Total 35 (delta 4), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:sayprachanh-lsvnv/study_2024-2025_arh-pc.git
  479fa13..9f1c2b4 master -> master
[sayprachanh@archlinux arch-pc]$
```

(Рис .4.24 Отправка изменений на сервер)

Затем я захожу на сайт GitHub, чтобы проверить правильность работы (Рис .4.25)

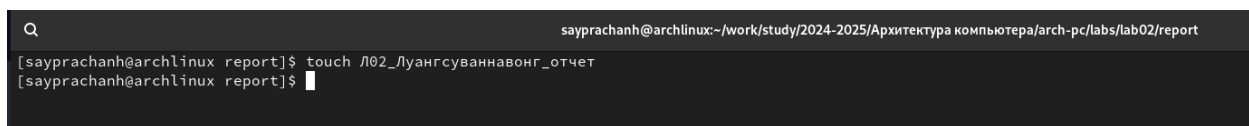


Name	Last commit message	Last commit date
..		
lab01	feat(main): make course structure	3 minutes ago
lab02	feat(main): make course structure	3 minutes ago
lab03	feat(main): make course structure	3 minutes ago
lab04	feat(main): make course structure	3 minutes ago
lab05	feat(main): make course structure	3 minutes ago
lab06	feat(main): make course structure	3 minutes ago
lab07	feat(main): make course structure	3 minutes ago
lab08	feat(main): make course structure	3 minutes ago
lab09	feat(main): make course structure	3 minutes ago
lab10	feat(main): make course structure	3 minutes ago
lab11	feat(main): make course structure	3 minutes ago
README.md	feat(main): make course structure	3 minutes ago
README.ru.md	feat(main): make course structure	3 minutes ago

(Рис .4.25 Страница репозитория)

## 5. Выполнение заданий для самостоятельной работы

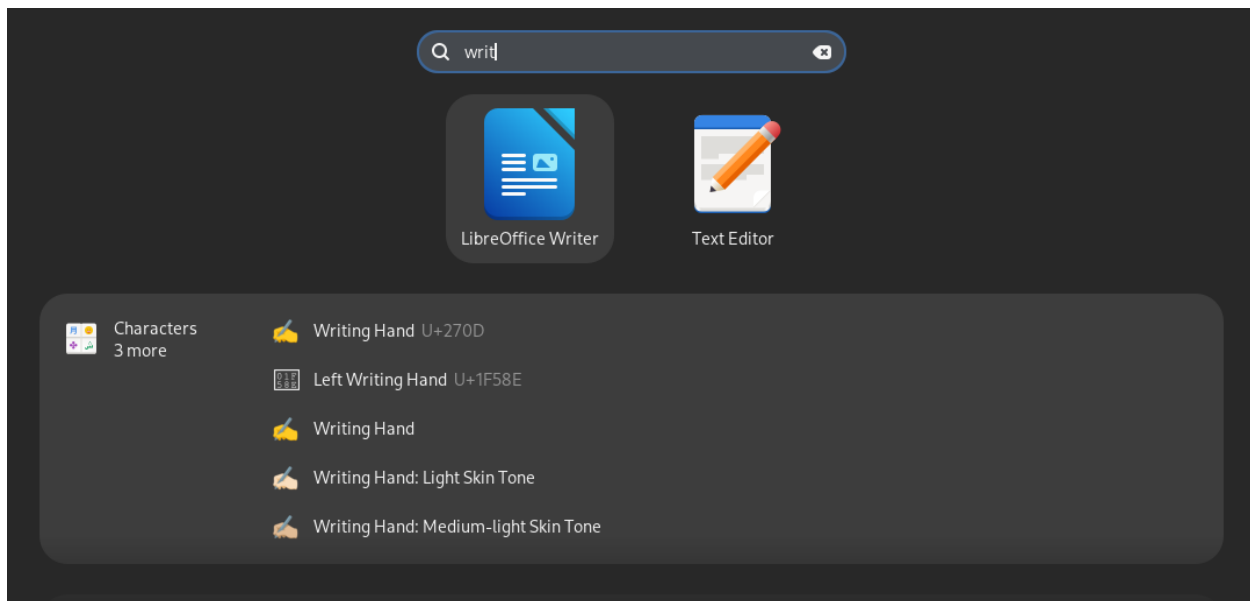
Сначала я перехожу в каталог `labs/lab02/report` с помощью команды `cd`, затем, используя `touch`, создаю файл отчета для второй лабораторной работы в каталоге (Рис .5.1)



```
sayprachanh@archlinux: ~/work/study/2024-2025/Архитектура компьютера/arch-pc/labs/lab02/report
[sayprachanh@archlinux report]$ touch /02_Луангсуваннавонг_отчет
[sayprachanh@archlinux report]$
```

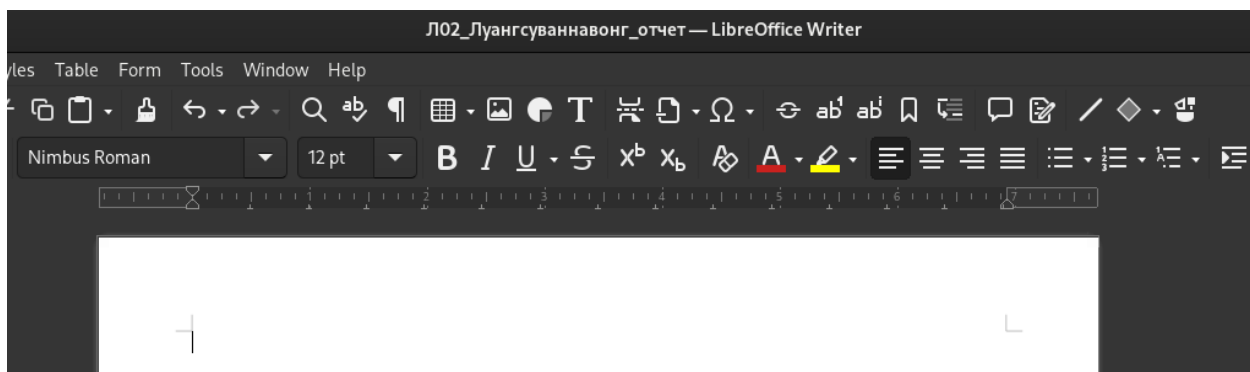
(Рис .5.1 Создание файла)

затем я ищу программу, которую буду использовать для написания отчета, в данном случае я использую LibreOffice Writer (Рис .5.2)



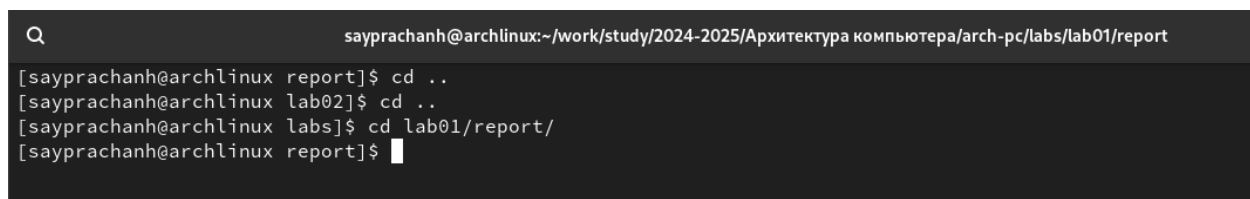
(Рис .5.2 Меню поиска)

После запуска программы, я открываю в ней файл и начинаю работать над отчетом (Рис .5.3)



(Рис .5.3 Работа с отчетом с программой LibreOffice Writer)

Я перехожу из подкаталога lab03/report в подкаталог lab01/report с помощью утилиты cd (Рис .5.4)



(Рис .5.4 Перемещение по каталогам)



Затем, используя команду `ls`, я проверяю наличие файла первой лабораторной работы, который должен находиться в подкаталоге `Downloads` домашнего каталога (Рис .5.5)

```
[sayprachanh@archlinux report]$ ls ~/Downloads
Л01_Луангсуваннавонг_отчет.pdf
[sayprachanh@archlinux report]$
```

(Рис .5.5 Проверка местонахождения файлов)

используя утилиту `cp`, я копирую файл первой лабораторной работы из подкаталога `Downloads` в подкаталог `lab1/report` и проверяю правильность выполнения команды с помощью команды `ls` (Рис .5.6)

```
[sayprachanh@archlinux report]$ cp ~/Downloads/Л01_Луангсуваннавонг_отчет.pdf /home/sayprachanh/work/study/2024-2025/'Архитектура компьютера'/arch-pc/labs/lab01/report
[sayprachanh@archlinux report]$ ls
bib image Makefile pandoc report.md Л01_Луангсуваннавонг_отчет.pdf
[sayprachanh@archlinux report]$
```

(Рис .5.6 Копирование файла)

добавляю файл в коммит, используя команду `git add` `Л01_Луангсуваннавонг_отчет.pdf` (Рис .5.7)

```
[sayprachanh@archlinux report]$ git add Л01_Луангсуваннавонг_отчет.pdf
[sayprachanh@archlinux report]$
```

(Рис .5.7 Добавление файла на сервер)

После этого я перехожу в репозиторий `lab02` и проделываю то же самое со вторым файлом лабораторной работы: Я добавляю файл с помощью `git add`, затем сохраняю изменения на сервере с помощью `git commit` (Рис .5.8)

```
[sayprachanh@archlinux report]$ git add Л02_Луангсуваннавонг_отчет
[sayprachanh@archlinux report]$ git commit -m "Adding file"
[master f3f3961] Adding file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 labs/lab02/report/Л02_Луангсуваннавонг_отчет
[sayprachanh@archlinux report]$
```

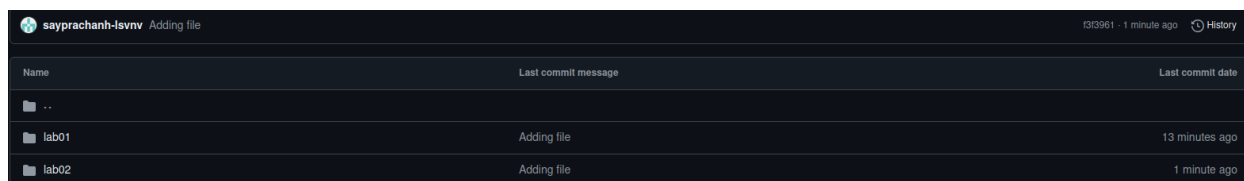
(Рис .5.8 Добавление и сохранение изменений на сервере)

Я отправляю сохраненные файлы изменений в основной репозиторий на сервере, используя команду `git push -f origin master` (Рис .5.9)

```
[sayprachanh@archlinux report]$ git push -f origin master
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 4 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 518 bytes | 518.00 KiB/s, done.
Total 5 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:sayprachanh-lsvnv/study_2024-2025_arh-pc.git
 544dd42..f3f3961 master -> master
[sayprachanh@archlinux report]$
```

(Рис .5.9 Загрузка файлов на сервер)

Захожу на сайт Github, чтобы проверить корректность работы, мы видим, что отображаются комментарии команды `commit` (Рис .5.10)

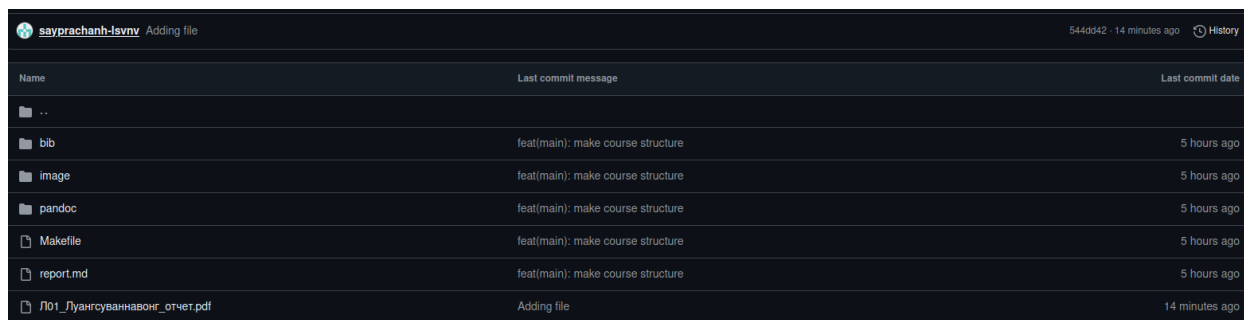


The screenshot shows a GitHub repository page for 'sayprachanh-lsvnv' with the title 'Adding file'. It displays a table of commit history with columns for Name, Last commit message, and Last commit date. The commits are for 'lab01' and 'lab02', both with the message 'Adding file'.

Name	Last commit message	Last commit date
..		
lab01	Adding file	13 minutes ago
lab02	Adding file	1 minute ago

(Рис .5.10 Страница каталога в репозитории)

Захожу в оба каталога, мы видим, что в каталоге `lab01/report` есть файл отчета о первой лабораторной работе, а также в каталоге `lab02/report`, в котором есть файл отчета о второй лабораторной работе (Рис .5.11 и Рис .5.12)



The screenshot shows a GitHub repository page for 'sayprachanh-lsvnv' with the title 'Adding file'. It displays a table of commit history with columns for Name, Last commit message, and Last commit date. The commits are for 'bib', 'image', 'pandoc', 'Makefile', 'report.md', and 'Л01\_Луангсуваннавонг\_отчет.pdf', all with the message 'feat(main): make course structure'.

Name	Last commit message	Last commit date
..		
bib	feat(main): make course structure	5 hours ago
image	feat(main): make course structure	5 hours ago
pandoc	feat(main): make course structure	5 hours ago
Makefile	feat(main): make course structure	5 hours ago
report.md	feat(main): make course structure	5 hours ago
Л01_Луангсуваннавонг_отчет.pdf	Adding file	14 minutes ago

(Рис .5.11 Каталог `lab01/report`)

sayprachanh-lsvnv Adding file			1313961 · 3 minutes ago	History
Name	Last commit message	Last commit date		
..				
bib	feat(main): make course structure	5 hours ago		
image	feat(main): make course structure	5 hours ago		
pandoc	feat(main): make course structure	5 hours ago		
Makefile	feat(main): make course structure	5 hours ago		
report.md	feat(main): make course structure	5 hours ago		
Л02_Луангсуваннавонг_отчет	Adding file	3 minutes ago		

(Рис .5.12 Каталог lab02/report)

## 6. Выводы

Во время этой лабораторной работы я изучил и понял идеологию и применение инструментов контроля версий, а также приобрел практические навыки работы с системой git.

## 7. Источники

1. [Архитектура ЭВМ](#)
2. [Git - Документация](#)