

Отчёт по лабораторной работе №2

Дисциплина: архитектура компьютера

Черная София Витальевна

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	14
4.6	Настройка каталога курса	17
4.7	Выполнение заданий для самостоятельной работы	20
5	Выводы	26
6	Список литературы	27

Список иллюстраций

4.1	Заполнение данных учетной записи GitHub	9
4.2	Аккаунт GitHub	9
4.3	Предварительная конфигурация git	10
4.4	Настройка кодировки	10
4.5	Создание имени для начальной ветки	11
4.6	Параметр autocrlf	11
4.7	Параметр safecrlf	11
4.8	Генерация SSH-ключа	12
4.9	Установка утилиты xclip	12
4.10	Копирование содержимого файла	13
4.11	Окно SSH and GPG keys	13
4.12	Добавление ключа	13
4.13	Создание рабочего пространства	14
4.14	Страница шаблона для репозитория	15
4.15	Окно создания репозитория	15
4.16	Созданный репозиторий	16
4.17	Перемещение между директориями	16
4.18	Клонирование репозитория	17
4.19	Окно с ссылкой для копирования репозитория	17
4.20	Перемещение между директориями	18
4.21	Удаление файлов	18
4.22	Создание каталогов	18
4.23	Добавление и сохранение изменений на сервере	19
4.24	Выгрузка изменений на сервер	19
4.25	Страница репозитория	20
4.26	Создание файла	20
4.27	Меню приложений	21
4.28	Работа с отчетом в текстовом процессоре	21
4.29	Перемещение между директориями	22
4.30	Проверка местонахождения файлов	22
4.31	Копирование файла	22
4.32	Добавление файла на сервер	22
4.33	Отправка в центральный репозиторий сохраненных изменений	23
4.34	Страница каталога в репозитории	23
4.35	Страница последних изменений в репозитории	24
4.36	Каталог lab01/report	24

4.37 Каталог lab02/report	25
-------------------------------------	----

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 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 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 заполнила основные данные учетной записи.

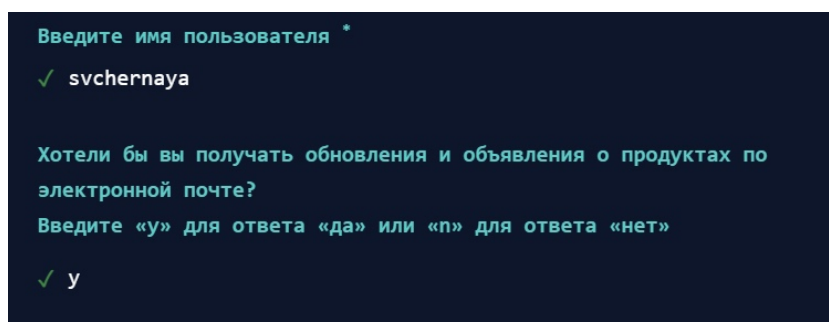
A dark-themed screenshot of the GitHub account creation form. The first field, labeled "Введите имя пользователя *" (Enter your username *), contains the text "svchernaya" with a green checkmark to its left. The second field, labeled "Хотели бы вы получать обновления и объявления о продуктах по электронной почте?" (Would you like to receive updates and product announcements by email?), contains the text "у" (yes) with a green checkmark to its left. Below the second field, there is a smaller instruction: "Введите «у» для ответа «да» или «н» для ответа «нет»" (Enter "y" for "yes" or "n" for "no").

Рис. 4.1: Заполнение данных учетной записи GitHub

Аккаунт создан (рис. [4.2]).

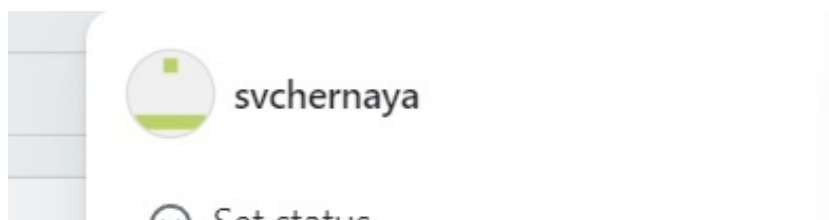


Рис. 4.2: Аккаунт GitHub

4.2 Базовая настройка Git

Открываю виртуальную машину, затем открываю терминал и делаю 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]).

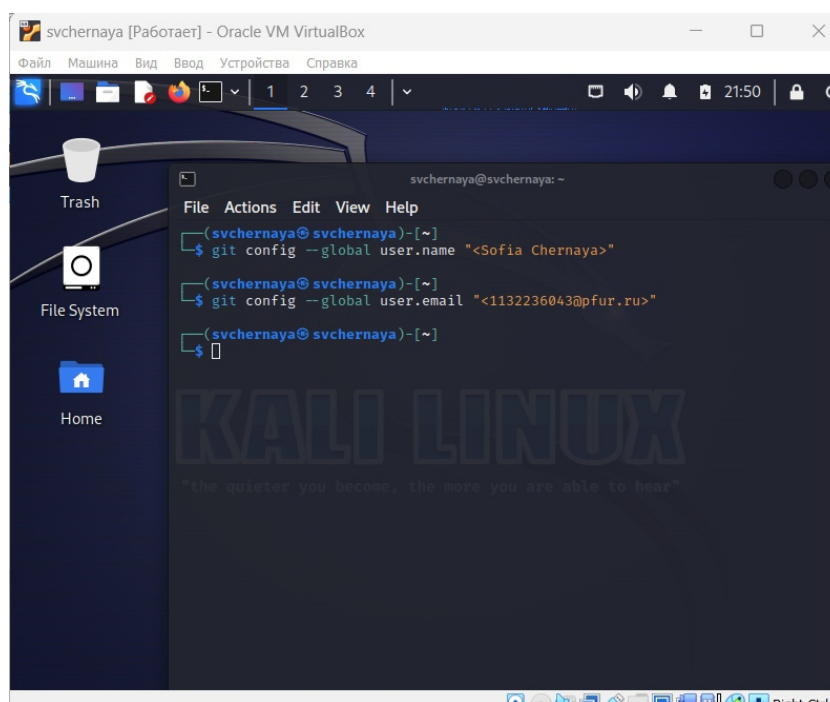


Рис. 4.3: Предварительная конфигурация git

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

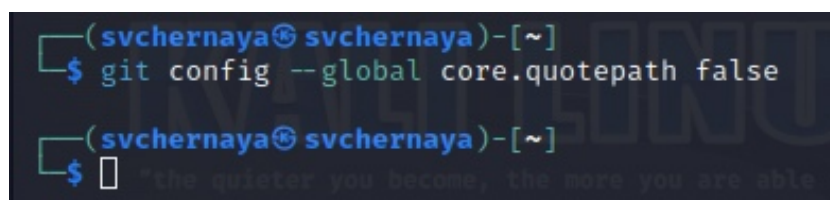
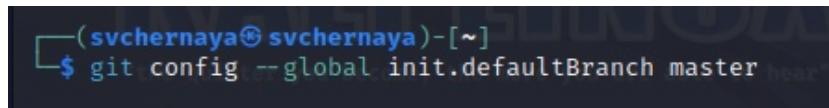


Рис. 4.4: Настройка кодировки

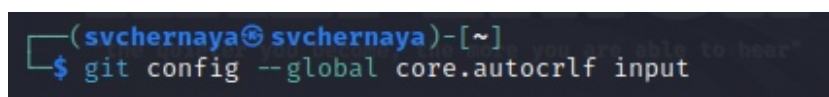
Задаю имя «master» для начальной ветки (рис. [4.5]).



```
(svchernaya@svchernaya)-[~]  
$ git config --global init.defaultBranch master
```

Рис. 4.5: Создание имени для начальной ветки

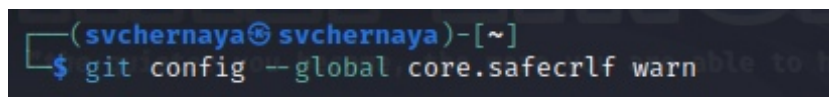
Задаю параметр `autocrlf` со значением `input`, так как я работаю в системе Linux, чтобы конвертировать CRLF в LF только при коммитах (рис. [4.6]). CR и LF – это символы, которые можно использовать для обозначения разрыва строки в текстовых файлах.



```
(svchernaya@svchernaya)-[~]  
$ 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.



```
(svchernaya@svchernaya)-[~]  
$ 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/`.

```
svchernaya@svchernaya: ~  
File Actions Edit View Help  
(svchernaya@svchernaya)-[~]  
$ ssh-keygen -C "Sofia Chernaya <1132236043@pfur.ru>"  
Generating public/private rsa key pair.  
Enter file in which to save the key (/home/svchernaya/.ssh/id_rsa):  
Created directory '/home/svchernaya/.ssh'.  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/svchernaya/.ssh/id_rsa  
Your public key has been saved in /home/svchernaya/.ssh/id_rsa.pub  
The key fingerprint is:  
SHA256:tHqY4aChRs3wdCyJ/Wv3J+l+oSrlr2ZqEM4uZJgK3atY Sofia Chernaya <1132236043@pfur.ru>  
The key's randomart image is:  
+--[RSA 3072]--+  
|  
|  o o  
| o = o .  
| * + . .  
| ..+... S  
|o..o oo= .  
|ooo. *=o+ + .  
|o+.E= *=o* o  
|=.o+ ..+*+*  
+--[SHA256]--+  
(svchernaya@svchernaya)-[~]  
$
```

Рис. 4.8: Генерация SSH-ключа

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

```
(svchernaya@svchernaya)-[~]  
$ sudo apt-get install -y xclip  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done  
The following NEW packages will be installed:  
  xclip  
0 upgraded, 1 newly installed, 0 to remove and 608 not upgraded.  
Need to get 23.3 kB of archives.  
After this operation, 65.5 kB of additional disk space will be used.  
Get:1 http://mirror.truenetwork.ru/kali kali-rolling/main amd64 xclip amd64 0.13-2 [23.3 kB]  
Fetched 23.3 kB in 5s (4969 B/s)  
Selecting previously unselected package xclip.  
(Reading database ... 398459 files and directories currently installed.)  
Preparing to unpack .../xclip_0.13-2_amd64.deb ...  
Unpacking xclip (0.13-2) ...  
Setting up xclip (0.13-2) ...  
Processing triggers for man-db (2.11.2-3) ...  
Processing triggers for kali-menu (2023.4.3) ...
```

Рис. 4.9: Установка утилиты xclip

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

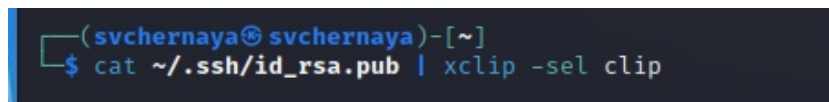


Рис. 4.10: Копирование содержимого файла

Открываю браузер, захожу на сайт GitHub. Открываю свой профиль и выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key» (рис. [4.11]).

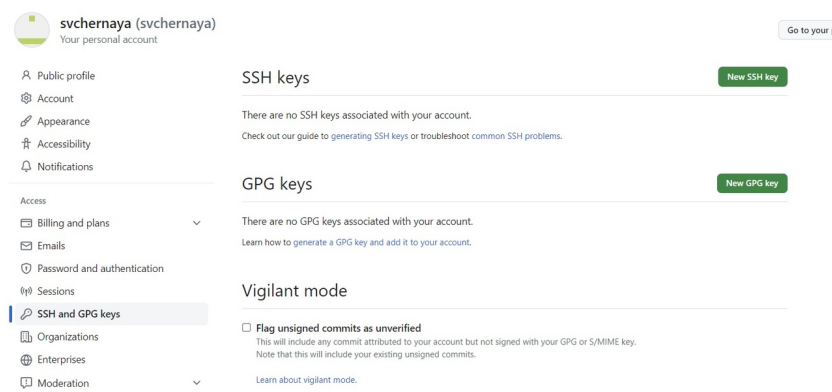


Рис. 4.11: Окно SSH and GPG keys

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

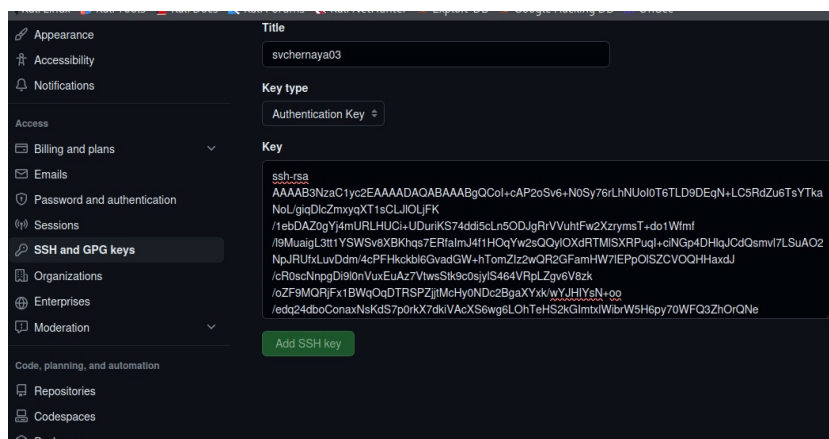
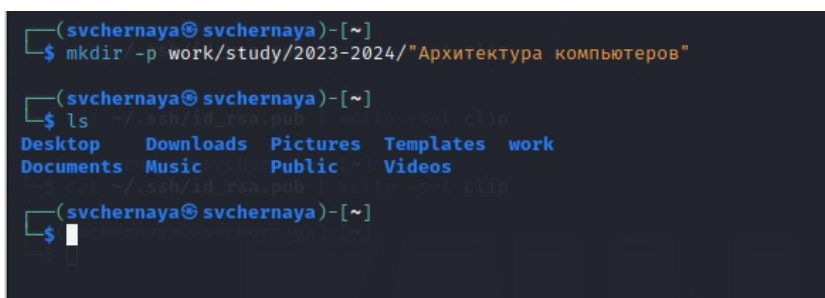


Рис. 4.12: Добавление ключа

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/«Архитектура компьютера»` рекурсивно. Далее проверяю с помощью `ls`, действительно ли были созданы необходимые мне каталоги (рис. [4.13]).



```
(svchernaya@svchernaya)~$ mkdir -p work/study/2023-2024/"Архитектура компьютеров"

(svchernaya@svchernaya)~$ ls
Desktop  Downloads  Pictures  Templates  work
Documents Music      Public    Videos
```

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/yamadharma/course-directory-student-template>. Далее выбираю «Use this template», чтобы использовать этот шаблон для своего репозитория (рис. [4.14]).

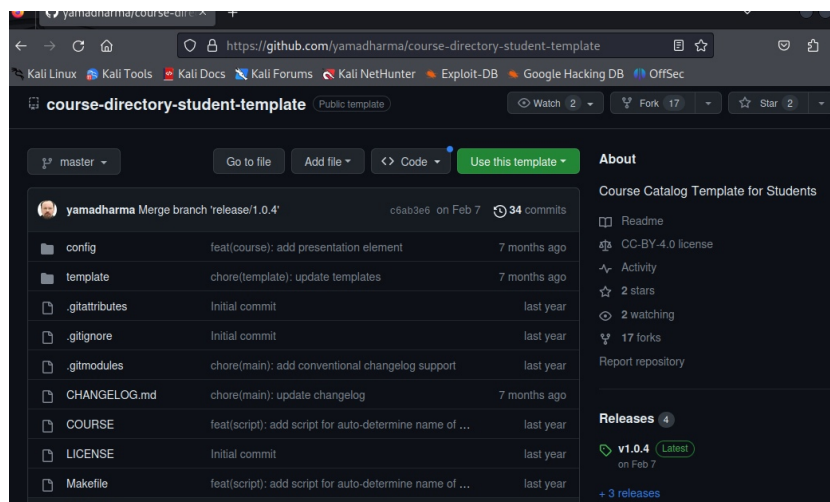


Рис. 4.14: Страница шаблона для репозитория

В открывшемся окне задаю имя репозитория (Repository name): study_2023–2024_arh-
pc и создаю репозиторий, нажимаю на кнопку «Create repository from template»
(рис. [4.15]).

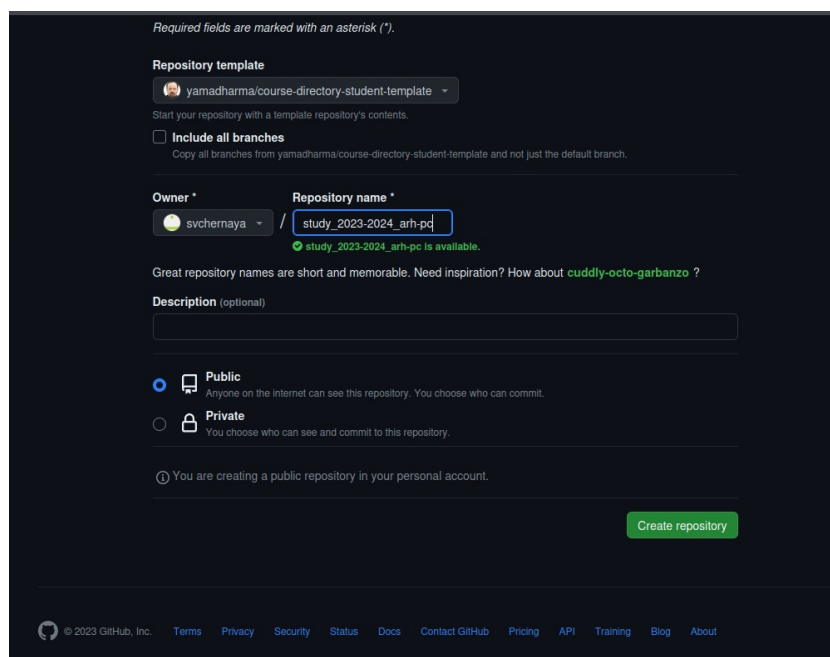


Рис. 4.15: Окно создания репозитория

Репозиторий создан (рис. [4.16]).

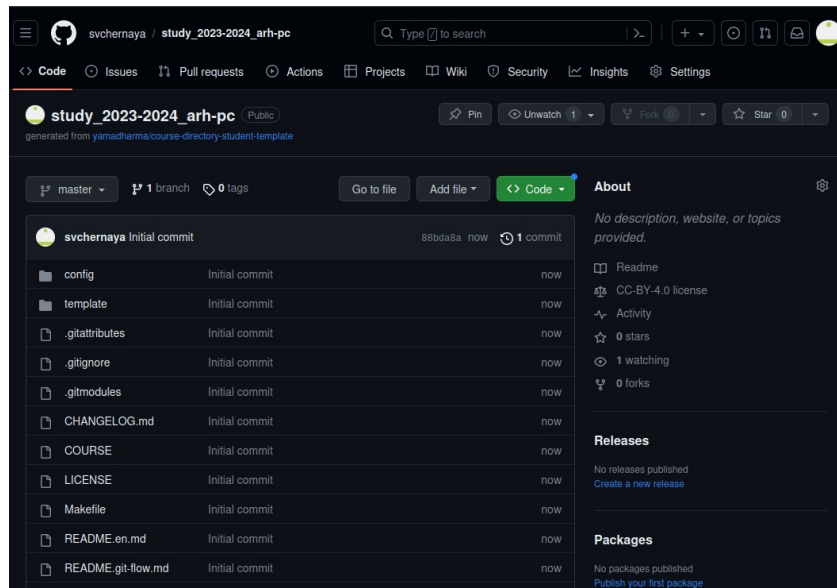


Рис. 4.16: Созданный репозиторий

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

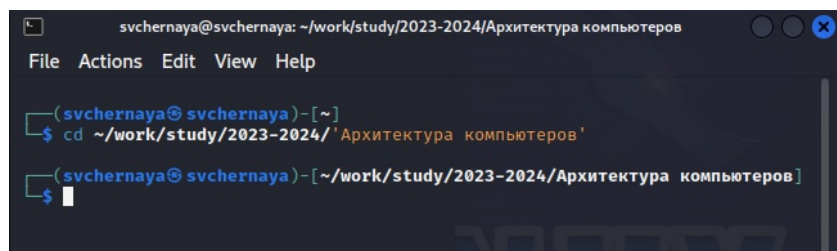
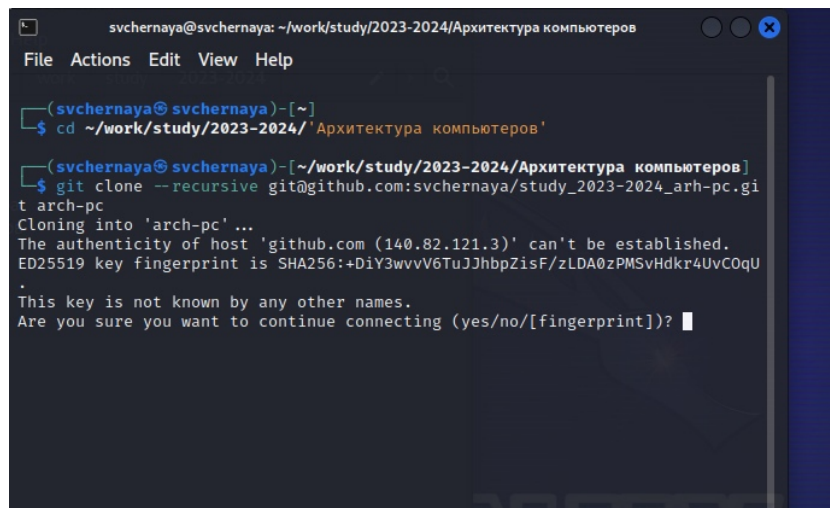


Рис. 4.17: Перемещение между директориями

Клонирую созданный репозиторий с помощью команды `git clone –recursive git@github.com:/study_2023–2024_arh-pc.git arch-pc` (рис. [4.18]).



```
svchernaya@svchernaya: ~/work/study/2023-2024/Архитектура компьютеров
File Actions Edit View Help

(svchernaya@svchernaya)-[~]
$ cd ~/work/study/2023-2024/ 'Архитектура компьютеров'

(svchernaya@svchernaya)-[~/work/study/2023-2024/Архитектура компьютеров]
$ git clone --recursive git@github.com:svchernaya/study_2023-2024_arh-pc.git
Cloning into 'arch-pc' ...
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpZisF/zLDA0zPMSvHdKr4UvC0Qu
.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

Рис. 4.18: Клонирование репозитория

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH» (рис. [4.19]).

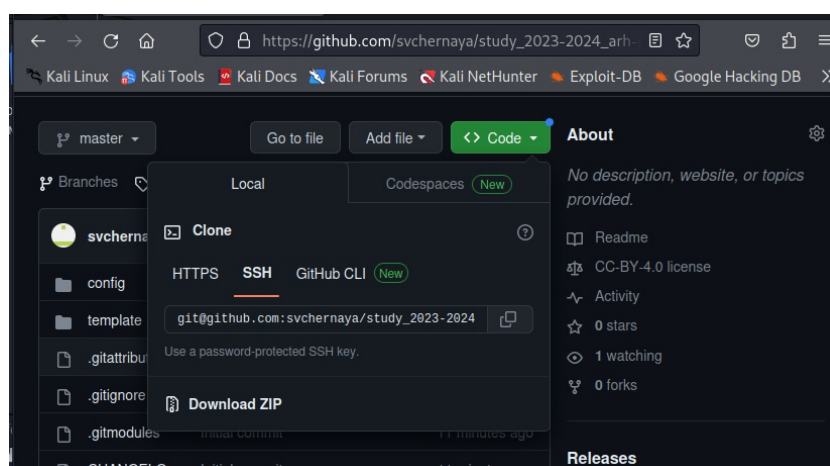


Рис. 4.19: Окно с ссылкой для копирования репозитория

4.6 Настройка каталога курса

Перехожу в каталог arch-рс с помощью утилиты cd (рис. [4.20]).

```
(svchernaya@svchernaya)~[~]
$ cd ~/work/study/2023-2024/Архитектура\ компьютеров/arch-pc
(svchernaya@svchernaya)~[~/study/2023-2024/Архитектура компьютеров/arch-pc]
$
```

Рис. 4.20: Перемещение между директориями

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

```
(svchernaya@svchernaya)~[~/study/2023-2024/Архитектура компьютеров/arch-pc]
$ rm package.json
(svchernaya@svchernaya)~[~/study/2023-2024/Архитектура компьютеров/arch-pc]
$
```

Рис. 4.21: Удаление файлов

Создаю необходимые каталоги (рис. [4.22]).

```
(svchernaya@svchernaya)~[~/study/2023-2024/Архитектура компьютеров/arch-pc]
$ echo arch-pc > COURSE
(svchernaya@svchernaya)~[~/study/2023-2024/Архитектура компьютеров/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]).

```
(svchernaya@svchernaya)~[~/../study/2023-2024/Архитектура компьютеров/arch-pc]
$ git add .

(svchernaya@svchernaya)~[~/../study/2023-2024/Архитектура компьютеров/arch-pc]
$ git commit -am 'feat(main): make courde structure'
[master 20068b0] feat(main): make courde 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
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/main.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 labs/lab02/report/report.md
create mode 100644 labs/lab03/presentation/Makefile
create mode 100644 labs/lab03/presentation/image/kulyabov.jpg
create mode 100644 labs/lab03/presentation/presentation.md
create mode 100644 labs/lab03/report/Makefile
create mode 100644 labs/lab03/report/bib/cite.bib
```

Рис. 4.23: Добавление и сохранение изменений на сервере

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

```
(svchernaya@svchernaya)~[~/../study/2023-2024/Архитектура компьютеров/arch-pc]
$ git push
Enumerating objects: 37, done.
Counting objects: 100% (37/37), done.
Delta compression using up to 2 threads
Compressing objects: 100% (29/29), done.
Writing objects: 100% (35/35), 342.14 KiB | 2.63 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:svchernaya/study_2023-2024_arh-pc.git
88bda8a..20068b0 master -> master
```

Рис. 4.24: Выгрузка изменений на сервер

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

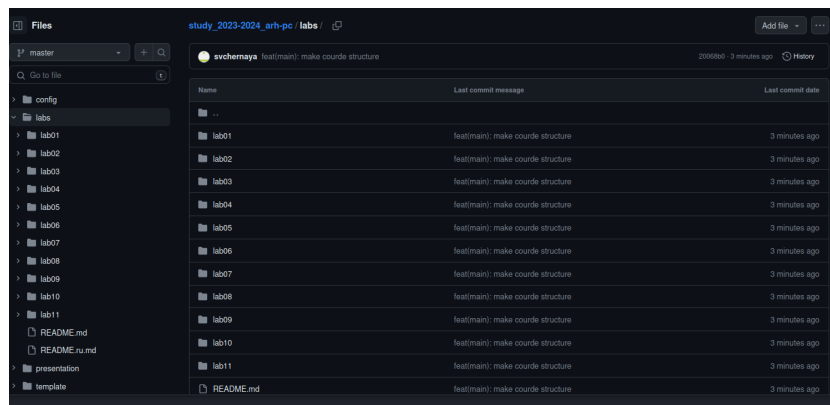


Рис. 4.25: Страница репозитория

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 второй лабораторной работе с помощью утилиты `touch` (рис. [4.27]).

```
(svchernaya@svchernaya) - [~/study/2023-2024/Архитектура компьютеров/arch-pc]
$ cd labs/lab03/report

(svchernaya@svchernaya) - [~/arch-pc/labs/lab03/report]
$ touch Л03_Черная_отчет
```

Рис. 4.26: Создание файла

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

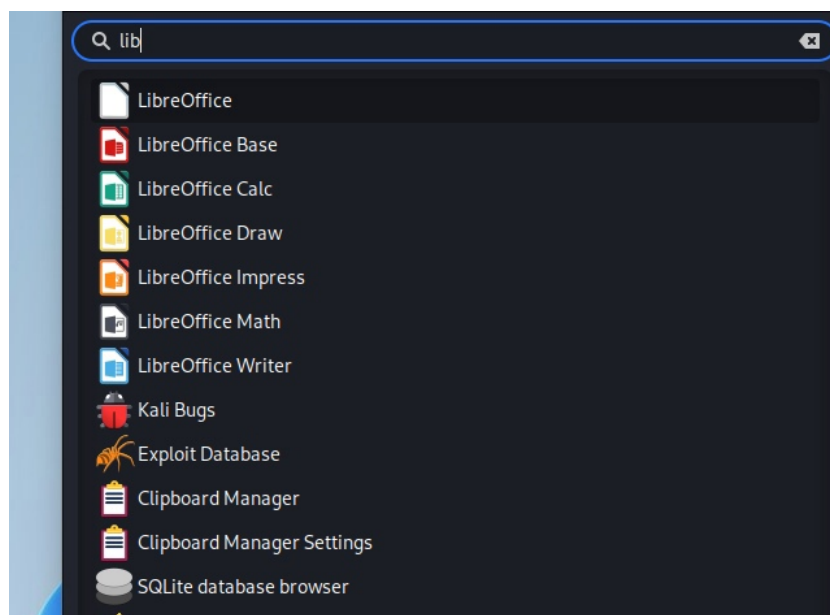


Рис. 4.27: Меню приложений

После открытия текстового процессора открываю в нем созданный файл и могу начать в нем работу над отчетом (рис. [??]).

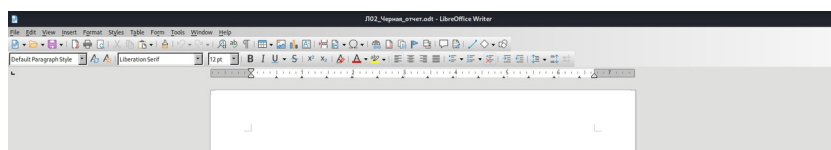


Рис. 4.28: Работа с отчетом в текстовом процессоре

2. Перехожу из подкаталога lab02/report в подкаталог lab01/report с помощью утилиты cd (рис. [4.29]).

```

(svchernaya@ svchernaya) - [~/../arch-pc/labs/lab02/report]
$ cd ..

(svchernaya@ svchernaya) - [~/../Архитектура компьютеров/arch-pc/labs/lab02]
$ cd ..

(svchernaya@ svchernaya) - [~/../2023-2024/Архитектура компьютеров/arch-pc/labs]
$ cd lab01/

(svchernaya@ svchernaya) - [~/../Архитектура компьютеров/arch-pc/labs/lab01]
$ cd ..

(svchernaya@ svchernaya) - [~/../2023-2024/Архитектура компьютеров/arch-pc/labs]
$ cd lab01/report

```

Рис. 4.29: Перемещение между директориями

Проверяю местонахождение файлов с отчетами по первой и второй лабораторным работам. Они должны быть в подкаталоге домашней директории «Загрузки», для проверки использую команду ls (рис. [4.30]).

```

(svchernaya@ svchernaya) - [~/../arch-pc/labs/lab01/report]
$ ls ~/Downloads
Л01_Черная_отчет.pdf

```

Рис. 4.30: Проверка местонахождения файлов

Копирую первую лабораторную с помощью утилиты cp и проверяю правильность выполнения команды cp с помощью ls (рис. [4.31]).

```

(svchernaya@ svchernaya) - [~/../arch-pc/labs/lab01/report]
$ cp ~/Downloads/Л01_Черная_отчет.pdf /home/svchernaya/work/study/2023-2024/Архитектура компьютеров/arch-pc/labs/lab01/report

(svchernaya@ svchernaya) - [~/../arch-pc/labs/lab01/report]
$ ls
Makefile  bib  image  pandoc  report.md  Л01_Черная_отчет.pdf

```

Рис. 4.31: Копирование файла

Добавляю файл(рис. [4.32]).

```

(svchernaya@ svchernaya) - [~/../arch-pc/labs/lab01/report]
$ git add Л01_Черная_отчет.pdf

```

Рис. 4.32: Добавление файла на сервер

Отправляю в центральный репозиторий сохраненные изменения командой git push -f origin master (рис. [4.33]).

```
(svchernaya@svchernaya)-[~/arch-pc/labs/lab02/report]
$ git push -f origin master
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 2 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 493 bytes | 493.00 KiB/s, done.
Total 5 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:svchernaya/study_2023-2024_arh-pc.git
de1dd6a..8ab011e master -> master
```

Рис. 4.33: Отправка в центральный репозиторий сохраненных изменений

Проверяю на сайте GitHub правильность выполнения заданий. Вижу, что пояснение к совершенным действиям отображается (рис. [4.34]).

svchernaya Add existing file 8ab011e · 2 minutes ago History		
Name	Last commit message	Last commit date
..		
lab01	Add existing file	24 minutes ago
lab02	Add existing file	2 minutes ago

Рис. 4.34: Страница каталога в репозитории

При просмотре изменений так же вижу, что были добавлены файлы с отчетами по лабораторным работам (рис. [4.35]).

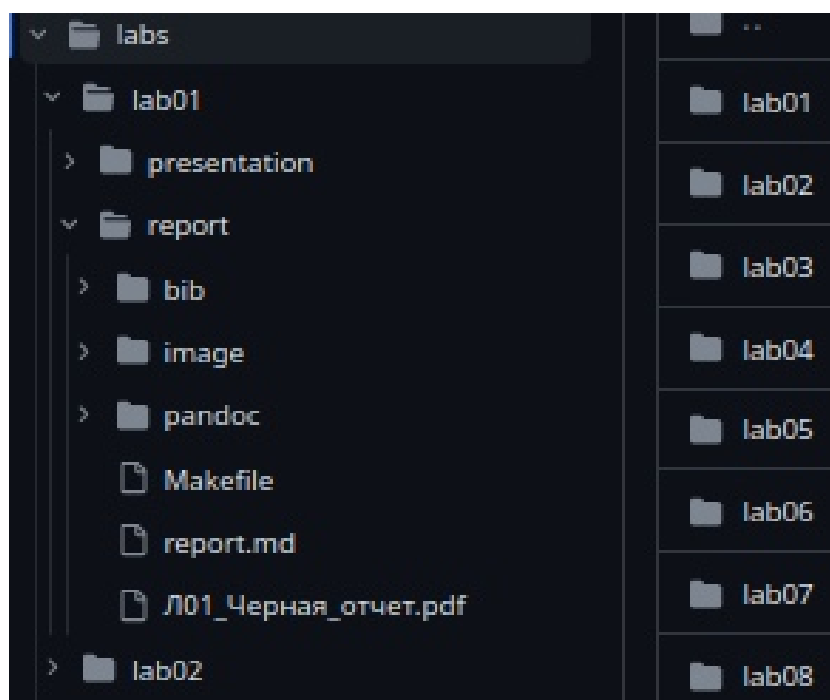


Рис. 4.35: Страница последних изменений в репозитории

Вижу, что отчеты по лабораторным работам находятся в соответствующих каталогах репозитория: отчет по первой - в lab01/report (рис. [4.36]), по второй – в lab02/report (рис. [4.37])

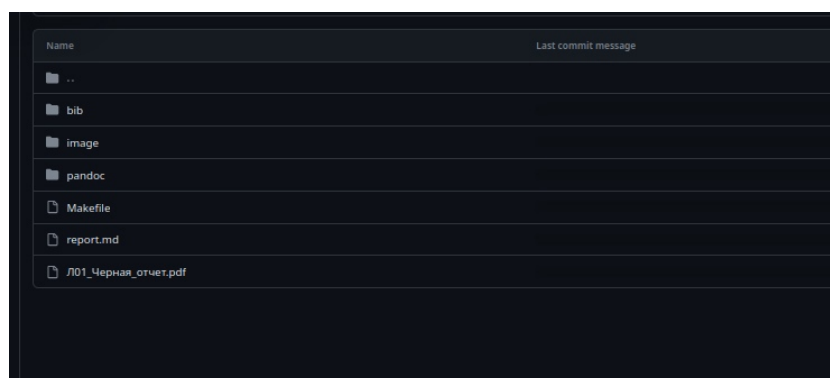


Рис. 4.36: Каталог lab01/report

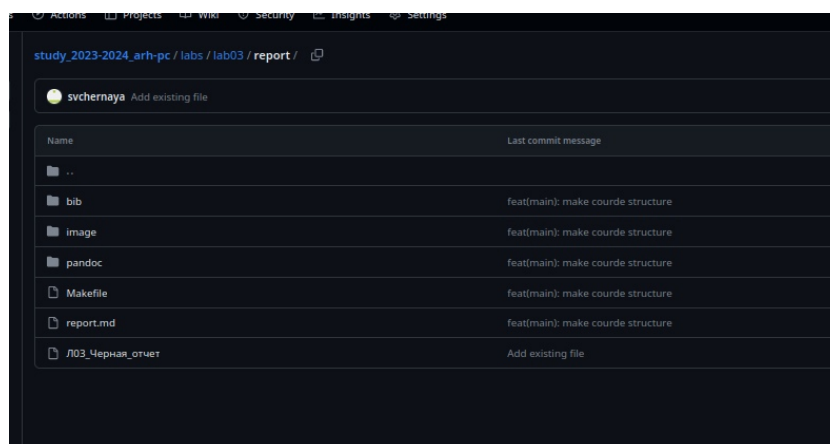


Рис. 4.37: Каталог lab02/report

5 Выводы

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

6 Список литературы

1. Архитектура ЭВМ