

Отчёт по лабораторной работе №3

Дисциплина: архитектура компьютера

Данил Берлов

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	13
4.5	Создание репозитория курса на основе шаблона	14
4.6	Настройка каталога курса	16
4.7	Выполнение заданий для самостоятельной работы	18
5	Выводы	23
6	Список литературы	24

Список иллюстраций

4.1	Заполнение данных учетной записи GitHub	9
4.2	Аккаунт GitHub	10
4.3	Предварительная конфигурация git	10
4.4	Настройка кодировки	10
4.5	Создание имени для начальной ветки	10
4.6	Параметр autocrlf	11
4.7	Параметр safecrlf	11
4.8	Генерация SSH-ключа	11
4.9	Установка утилиты xclip	12
4.10	Копирование содержимого файла	12
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	14
4.15	Окно создания репозитория	15
4.16	Созданный репозиторий	15
4.17	Перемещение между директориями	15
4.18	Клонирование репозитория	16
4.19	Окно с ссылкой для копирования репозитория	16
4.20	Перемещение между директориями	16
4.21	Удаление файлов	16
4.22	Создание каталогов	17
4.23	Добавление и сохранение изменений на сервере	17
4.24	Выгрузка изменений на сервер	17
4.25	Страница репозитория	18
4.26	Создание файла	18
4.27	Меню приложений	18
4.28	Работа с отчетом в текстовом процессоре	19
4.29	Перемещение между директориями	19
4.30	Проверка местонахождения файлов	19
4.31	Копирование файла	19
4.32	Перемещение между директориями	19
4.33	Копирование файла	20
4.34	Добавление файла на сервер	20
4.35	Перемещение между директориями	20
4.36	Добавление файла на сервер	20
4.37	Подкаталоги и файлы в репозитории	20

4.38 Отправка в центральный репозиторий сохраненных изменений .	21
4.39 Страница каталога в репозитории	21
4.40 Страница последних изменений в репозитории	21
4.41 Каталог lab01/report	22
4.42 Каталог lab02/report	22
4.43 Каталог lab03/report	22

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.

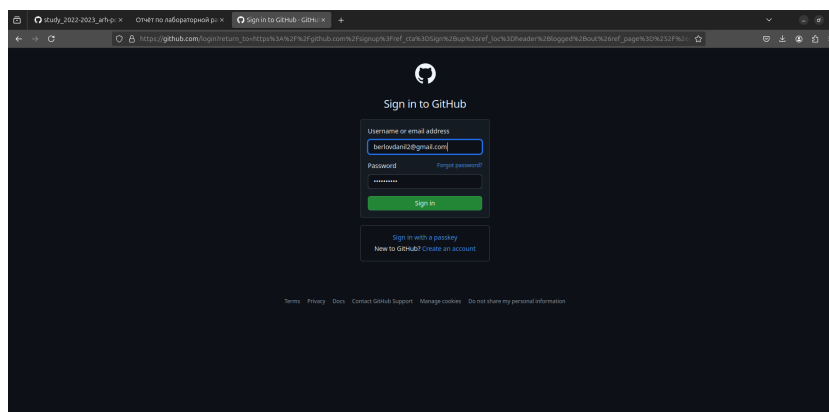


Рис. 4.1: Заполнение данных учетной записи GitHub

Аккаунт создан (рис. 4.2).

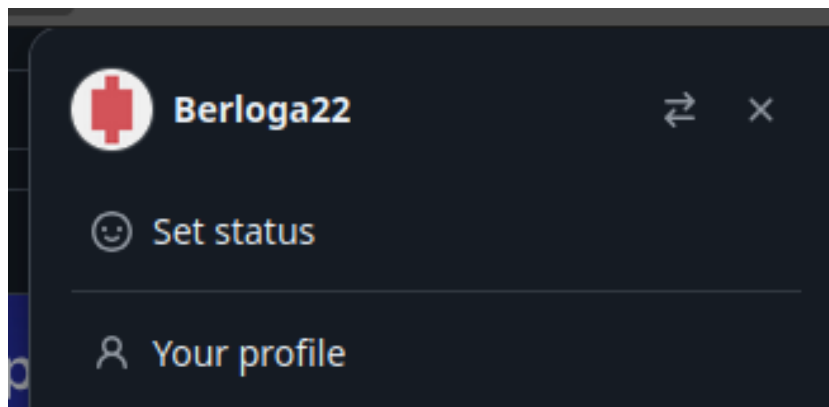


Рис. 4.2: Аккаунт GitHub

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).

```
berloga@berloga-VMware-Virtual-Platform:~$ git config --global user.name "<Danil Berlov>"
berloga@berloga-VMware-Virtual-Platform:~$ git config --global user.email "<1032244490@pfur.ru>"
```

Рис. 4.3: Предварительная конфигурация git

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

```
berloga@berloga-VMware-Virtual-Platform:~$ git config --global core.quotePath false
berloga@berloga-VMware-Virtual-Platform:~$
```

Рис. 4.4: Настройка кодировки

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

```
berloga@berloga-VMware-Virtual-Platform:~$ git config --global init.defaultBranch master
berloga@berloga-VMware-Virtual-Platform:~$
```

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 сим-

волы, которые можно использовать для обозначения разрыва строки в текстовых файлах.

```
berloga@berloga-VMware-Virtual-Platform:~$ git config --global core.autocrlf input
berloga@berloga-VMware-Virtual-Platform:~$
```

Рис. 4.6: Параметр autocrlf

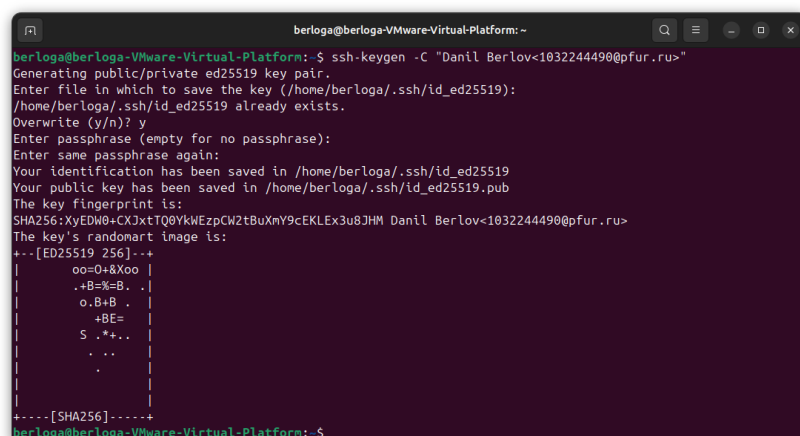
Задаю параметр safecrlf со значением warn, так Git будет проверять преобразование на обратимость (рис. 4.7). При значении warn Git только выведет предупреждение, но будет принимать необратимые конвертации.

```
berloga@berloga-VMware-Virtual-Platform:~$ git config --global core.safecrlf warn
berloga@berloga-VMware-Virtual-Platform:~$
```

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/`.



```
berloga@berloga-VMware-Virtual-Platform:~$ ssh-keygen -C "Danil Berlov<1032244490@pfur.ru>"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/berloga/.ssh/id_ed25519):
/home/berloga/.ssh/id_ed25519 already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/berloga/.ssh/id_ed25519
Your public key has been saved in /home/berloga/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:XYEDW0+CXJxtTQ0YkWEzpcW2tBuXmY9cEKLEx3u8JHM Danil Berlov<1032244490@pfur.ru>
The key's randomart image is:
+--[ED25519 256]--+
|      oo=O+&Xoo   |
|    .+B=%-B. .   |
|   o.B+B .       |
|    +BE=         |
|   S .*+..       |
|    .  .         |
|    .            |
+-----[SHA256]-----+
berloga@berloga-VMware-Virtual-Platform:~$
```

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 -y от имени суперпользователя, введя в начале команды sudo (рис. 4.9).

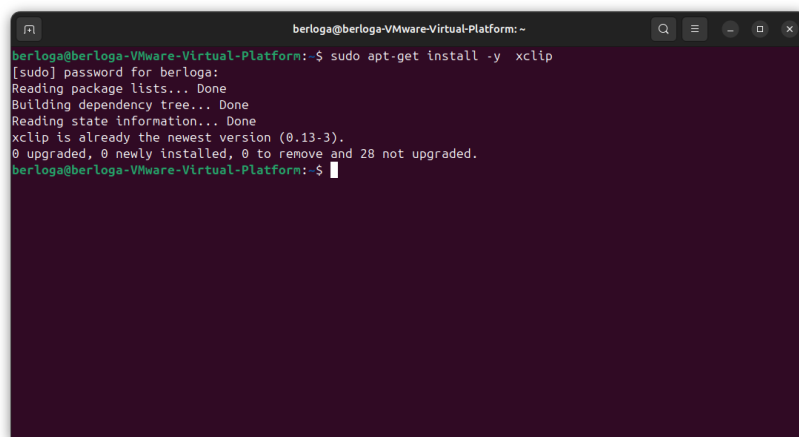
A terminal window with a dark purple background. The prompt is 'berloga@berloga-VMware-Virtual-Platform: ~'. The command 'sudo apt-get install -y xclip' is entered. The output shows the password prompt, package lists being read, dependency tree being built, and state information being read. It then states 'xclip is already the newest version (0.13-3)' and '0 upgraded, 0 newly installed, 0 to remove and 28 not upgraded.' The prompt returns to 'berloga@berloga-VMware-Virtual-Platform: ~\$'.

Рис. 4.9: Установка утилиты xclip

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

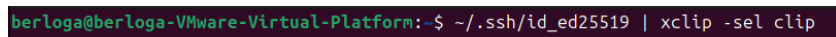
A terminal window with a dark purple background. The prompt is 'berloga@berloga-VMware-Virtual-Platform: ~\$'. The command '~/.ssh/id_ed25519 | xclip -sel clip' is entered.

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).

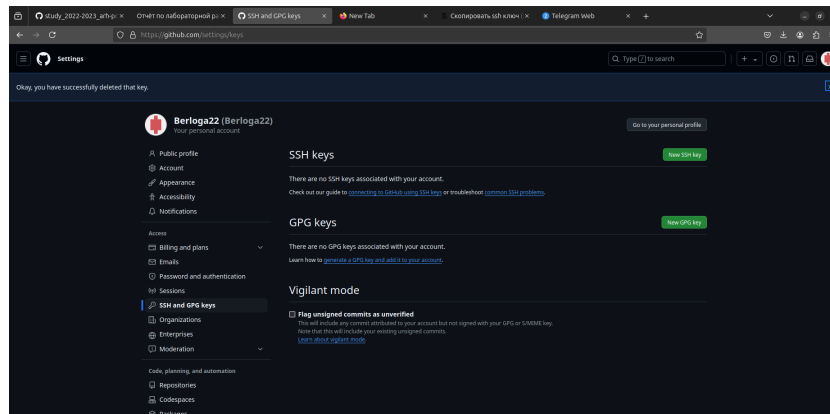


Рис. 4.11: Окно SSH and GPG keys

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

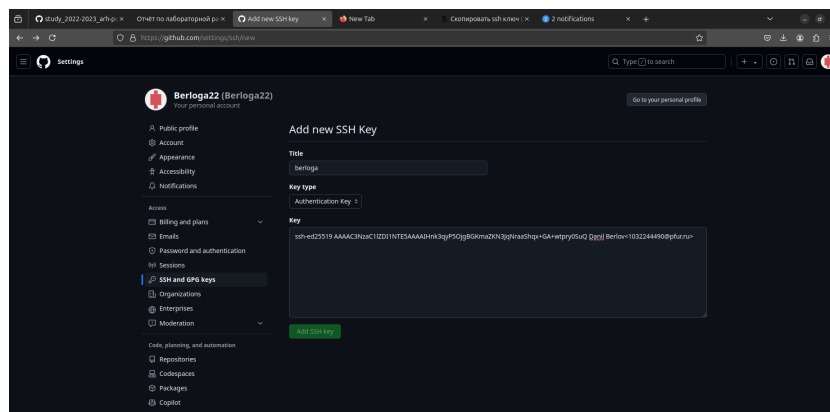


Рис. 4.12: Добавление ключа

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/2022-2023/“Архитектура компьютера” рекурсивно. Далее проверяю с помощью ls, действительно ли были созданы необходимые мне каталоги (рис. 4.13).

```
berloga@berloga-VMware-Virtual-Platform:~$ mkdir -p ~/work/study/2024-2025/"Computer architecture"
berloga@berloga-VMware-Virtual-Platform:~$ 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).

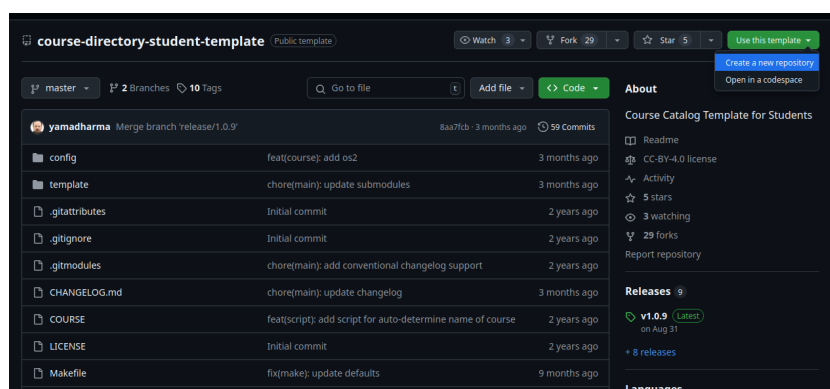


Рис. 4.14: Страница шаблона для репозитория

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

Рис. 4.15: Окно создания репозитория

Репозиторий создан (рис. 4.16).

File	Commit	Time
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
Makefile	Initial commit	now
README-en.md	Initial commit	now

Рис. 4.16: Созданный репозиторий

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

```
berloga@berloga-Virtual-Platform:~$ cd ~/work/study/2024-2025/"Computer architecture"
berloga@berloga-Virtual-Platform:~/work/study/2024-2025/Computer architecture$
```

Рис. 4.17: Перемещение между директориями

Клонирую созданный репозиторий с помощью команды `git clone --recursive git@github.com:/study_2022–2023_arh-pc.git arch-pc` (рис. 4.18).

```
berloga@berloga-Virtual-Platform: ~/work/study/2024-2025/Computer_architecture/arch-pc $ git clone --recursive git@github.com:Berloga22/study_2024-2025_arch-pc.git arch-pc
```

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).

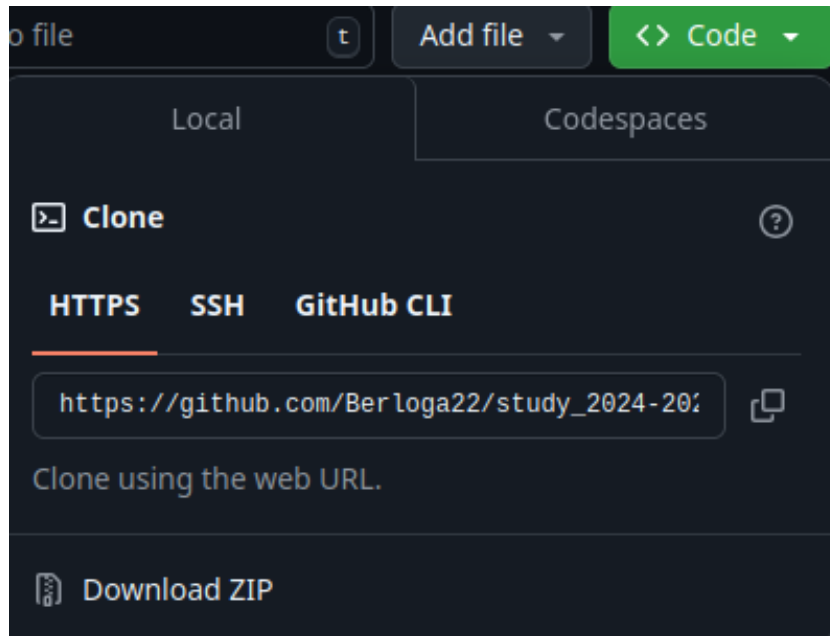


Рис. 4.19: Окно с ссылкой для копирования репозитория

4.6 Настройка каталога курса

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

```
berloga@berloga-Virtual-Platform: ~/work/study/2024-2025/Computer_architecture $ cd ~/work/study/2024-2025/Computer_architecture/arch-pc
berloga@berloga-Virtual-Platform: ~/work/study/2024-2025/Computer_architecture/arch-pc $
```

Рис. 4.20: Перемещение между директориями

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

```
berloga@berloga-Virtual-Platform: ~/work/study/2024-2025/Computer_architecture/arch-pc $ rm package.json
```

Рис. 4.21: Удаление файлов

Создаю необходимые каталоги (рис. 4.22).


```
berloga@Berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc$ echo arch-pc > COURSE
berloga@Berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc$ make
mkdir
```

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).

```
berloga@Berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc$ git add .
berloga@Berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc$ git commit -m 'feat(main): make course structure'
[master 4f5cae4] feat(main): make course structure
223 files changed, 53681 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/lab01/presentation/projectile
create mode 100644 labs/lab01/presentation/textlabroot
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/figure/placement_880_680_tech.jpg
create mode 100644 labs/lab01/report/pandoc/csl/gost-r-7-8-5-2008-numeric.csl
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_secnos.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_lablenos.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/Makefile
```

Рис. 4.23: Добавление и сохранение изменений на сервере

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

```
berloga@Berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/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), 341.27 KiB | 2.71 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:Berloga22/study_2024-2025_arh-pc.git
7d3d9a1..4f5cae4 master -> master
```

Рис. 4.24: Выгрузка изменений на сервер

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

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

Рис. 4.25: Страница репозитория

4.7 Выполнение заданий для самостоятельной работы

1. Перехожу в директорию `labs/lab03/report` с помощью утилиты `cd`. Создаю в каталоге файл для отчета по третьей лабораторной работе с помощью утилиты `touch` (рис. 4.26).

```
berloga@berloga-Virtual-Platform: $ cd work/study/2024-2025/"Computer architecture"/arch-pc/labs/lab03/report
```

Рис. 4.26: Создание файла

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

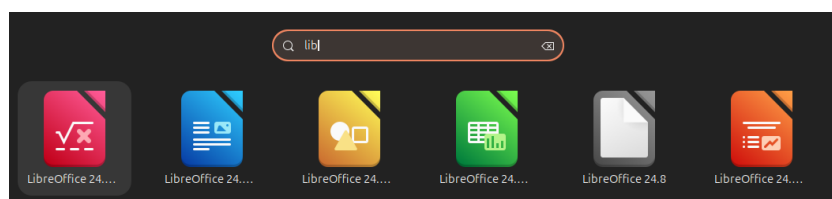


Рис. 4.27: Меню приложений

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

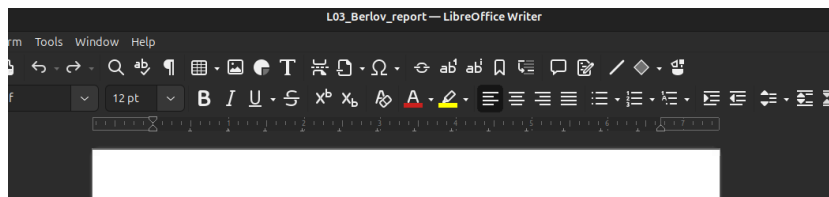


Рис. 4.28: Работа с отчетом в текстовом процессоре

2. Перехожу из подкаталога lab03/report в подкаталог lab01/report с помощью утилиты cd (рис. 4.29).

```
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs/lab03/report$ cd ..
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs/lab03$ cd ..
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs$ cd lab01/report/
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs/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).

```
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs/lab01/report$ ls ~/Downloads
LibreOffice_24.2.7_Linux_x86-64_deb  LibreOffice_24.2.7_Linux_x86-64_deb.tar.gz  Л01_Берлов_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).

```
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs/lab01/report$ cp ~/Downloads/L01_Sepnoa_otchet.pdf /home/berloga/work/
study/2024-2025/Computer architecture/arch-pc/labs/lab01/report
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs/lab01/report$ ls
lib  images  Makefile  README  report.md  Л01_Sepnoa_otchet.pdf
```

Рис. 4.31: Копирование файла

Перехожу из подкаталога lab01/report в подкаталог lab02/report с помощью утилиты cd (рис. 4.32).

```
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs/lab01/report$ cd ..
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs/lab01$ cd ..
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs$ cd lab02/report
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer architecture/arch-pc/labs/lab02/report$
```

Рис. 4.32: Перемещение между директориями

Копирую вторую лабораторную с помощью утилиты `cp` и проверяю правильность выполнения команды `cp` с помощью `ls` (рис. 4.33).

```
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer_architecture/arch-pc/labs/lab02/report$ cd .. ; cd .. ; cd lab01/report/
```

Рис. 4.33: Копирование файла

3. Добавляю с помощью команды `git add` в коммит созданные файлы: Л02_Дворкина отчет (рис. 4.34). и

```
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer_architecture/arch-pc/labs/lab02/report$ git add /01_Берлов_отчет.pdf
```

Рис. 4.34: Добавление файла на сервер

Перехожу в директорию, в которой находится отчет по первой лабораторной работе с помощью `cd` (рис. 4.35).

```
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer_architecture/arch-pc/labs/lab02/report$ touch L03_Berlov_report
```

Рис. 4.35: Перемещение между директориями

Добавляю файл Л01_Дворкина_отчет (рис. 4.36).

```
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer_architecture/arch-pc/labs/lab02/report$ git add L02_Berlov_report
```

Рис. 4.36: Добавление файла на сервер

Сохраняю изменения на сервере командой `git commit -m "..."`, поясняя, что добавила файлы.

То же самое делаю для отчета по третьей лабораторной работе: перехожу в директорию `labs/lab03/report` с помощью `cd`, добавляю с помощью `git add` нужный файл, сохраняю изменения с помощью `git commit` (рис. 4.37).

```
berloga@berloga-Virtual-Platform: /work/study/2024-2025/Computer_architecture/arch-pc/labs/lab02/report$ git commit -m "ADD existing file"
[master 0d58740] ADD existing file
2 files changed, 0 insertions(+), 0 deletions(-)
create mode 100644 labs/lab01/report/01_Berlov_отчет.pdf
create mode 100644 labs/lab02/report/L02_Berlov_report
```

Рис. 4.37: Подкаталоги и файлы в репозитории

Отправляю в центральный репозиторий сохраненные изменения командой `git push -f origin master` (рис. 4.38).

```

Berloga@Berloga-Virtual-Platform: /work/study/2024-2023/Computer_Architecture/arh-pc/labs/lab02/reports$ git push -f origin master
Enumerating objects: 14, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 4 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 1.47 MiB | 2.18 MiB/s, done.
Total 8 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
remote: This repository moved. Please use the new location:
remote: git@github.com:Berloga22/study_2024-2023_arh-pc.git
To github.com:Berloga22/study_2024-2023_arh-pc.git
 4f5cae4..0d58749  master -> master

```

Рис. 4.38: Отправка в центральный репозиторий сохраненных изменений

Проверяю на сайте GitHub правильность выполнения заданий. Вижу, что пояснение к совершенным действиям отображается (рис. 4.39).

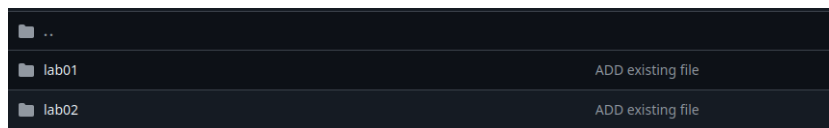


Рис. 4.39: Страница каталога в репозитории

При просмотре изменений так же вижу, что были добавлены файлы с отчетами по лабораторным работам (рис. 4.40).

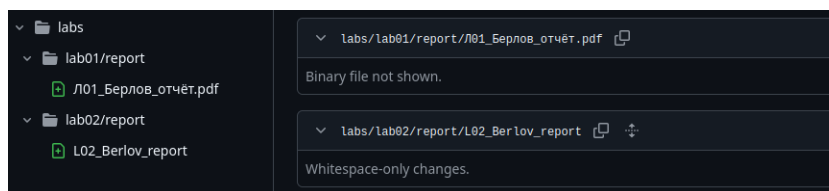


Рис. 4.40: Страница последних изменений в репозитории

Вижу, что отчеты по лабораторным работам находятся в соответствующих каталогах репозитория: отчет по первой - в lab01/report (рис. 4.41), по второй – в lab02/report (рис. 4.42), по третьей в - lab03/report (рис. 4.43).

study_2024-2025_arh-pc / labs / lab01 / report /

Danil Berlov ADD existing file

Name	Last commit message
..	
bib	feat(main): make course structure
image	feat(main): make course structure
pandoc	feat(main): make course structure
Makefile	feat(main): make course structure
report.md	feat(main): make course structure
L01_Берлов_отчёт.pdf	ADD existing file

Рис. 4.41: Каталог lab01/report

study_2024-2025_arh-pc / labs / lab02 / report /

Danil Berlov ADD existing file

Name	Last commit message
..	
bib	feat(main): make course structure
image	feat(main): make course structure
pandoc	feat(main): make course structure
L02_Berlov_report	ADD existing file
Makefile	feat(main): make course structure
report.md	feat(main): make course structure

Рис. 4.42: Каталог lab02/report

Danil Berlov Add files 5743561 · 29 minutes ago

Name	Last commit message	Last commit da...
..		
bib	feat(main): make course structure	last month
image	Add files	29 minutes ago
pandoc	feat(main): make course structure	last month
Makefile	feat(main): make course structure	last month

Рис. 4.43: Каталог lab03/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
2. Git - gitattributes Документация