

Отчёт по лабораторной работе №2

Дисциплина: архитектура компьютера

Корчагин Алексей Павлович нмм-бд 02-23

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

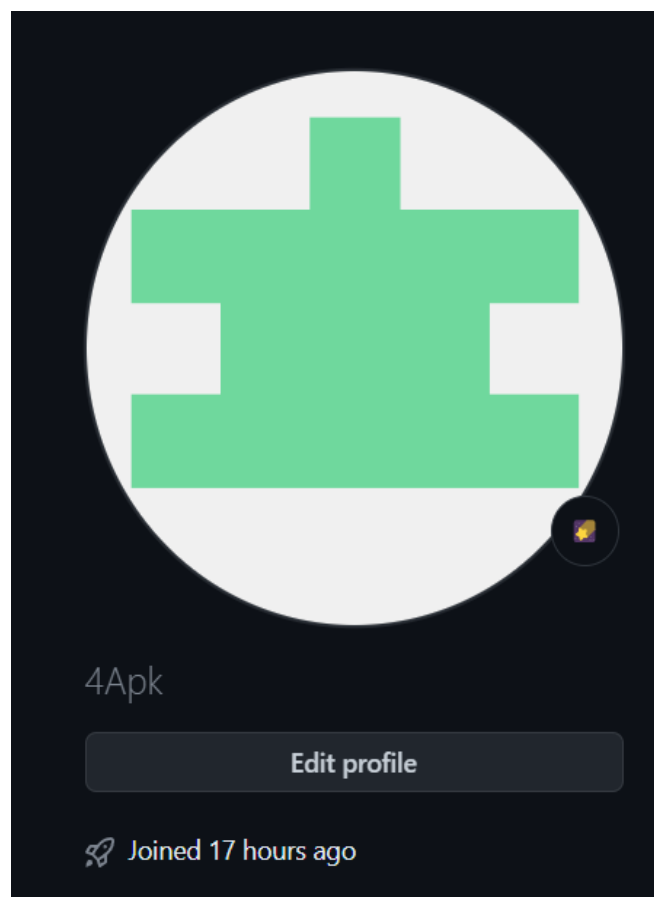
другим пользователям получить рабочую копию или препятствует изменению рабочей копии файла средствами файловой системы ОС, обеспечивая таким образом привилегированный доступ только одному пользователю, работающему с файлом. Системы контроля версий также могут обеспечивать дополнительные, более гибкие 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 Создание аккаунта GitHub

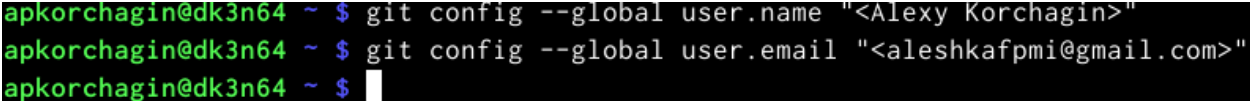
Создал аккаунт

Рис. 4.1:



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).



```
apkorchagin@dk3n64 ~ $ git config --global user.name "<Alexy Korchagin>"
apkorchagin@dk3n64 ~ $ git config --global user.email "<aleshkafpmi@gmail.com>"
apkorchagin@dk3n64 ~ $
```

Рис. 4.2: Предварительная конфигурация git

Настраиваю utf-8 в выводе сообщений git для корректного отображения символов (рис. 4.2).


```
apkorchagin@dk3n64 ~ $ git config --global core.quotePath false
apkorchagin@dk3n64 ~ $
```

Рис. 4.3: Настройка кодировки

Задаю имя «master» для начальной ветки (рис. 4.4).

```
apkorchagin@dk3n64 ~ $ git config --global init.defaultBranch master
apkorchagin@dk3n64 ~ $
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report $ git commit -m "Add existing Files"
[master 63ea025] Add existing Files
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100755 labs/lab01/report/Л01_Корчагин_отчет.doc
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report $
```

Рис. 4.4: 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.

```
apkorchagin@dk3n64 ~ $ git config --global core.autocrlf input
apkorchagin@dk3n64 ~ $
```

Рис. 4.5: Параметр autocrlf

Задаю параметр safecrlf со значением warn, так Git будет проверять преобразование на обратимость (рис. 4.6). При значении warn Git только выведет предупреждение, но будет принимать необратимые конвертации.

```
apkorchagin@dk3n64 ~ $ git config --global core.safecrlf warn
apkorchagin@dk3n64 ~ $
```

Рис. 4.6: 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. 4.7). Ключ автоматически сохранится в каталоге `~/.ssh/`.

```
apkorchagin@dk3n64 ~ $ ssh-keygen -C "Alexy Korchagin <aleshkafpmi@gmail.com>"
Generating public/private rsa key pair.
Enter file in which to save the key (/afs/.dk.sci.pfu.edu.ru/home/a/p/apkorchagin/.ssh/id_rsa):
/afs/.dk.sci.pfu.edu.ru/home/a/p/apkorchagin/.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 /afs/.dk.sci.pfu.edu.ru/home/a/p/apkorchagin/.ssh/id_rsa
Your public key has been saved in /afs/.dk.sci.pfu.edu.ru/home/a/p/apkorchagin/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:S4I+fvi1BKdley1fQEjYu2tfHLVBsjwNnJgXetkb1Ec Alexy Korchagin <aleshkafpmi@gmail.com>
The key's randomart image is:
+---[RSA 3072]-----+
|      +==+E|
|      .o==+o+|
|      .o0==+|
|      o  o*o.B|
|      S . ...=|
|      o . . . .|
|      o . .o o|
|      o. +. o |
|      .==...|
+----[SHA256]-----+
apkorchagin@dk3n64 ~ $
```

Рис. 4.7: Генерация SSH-ключа

```
apkorchagin@dk3n64 ~ $ cat ~/.ssh/id_rsa.pub | xclip -sel clip
apkorchagin@dk3n64 ~ $
```

Рис. 4.8: Копирование ключа из консоли

Открываю браузер, захожу на сайт GitHub. Открываю свой профиль и выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key» (рис. 4.9).

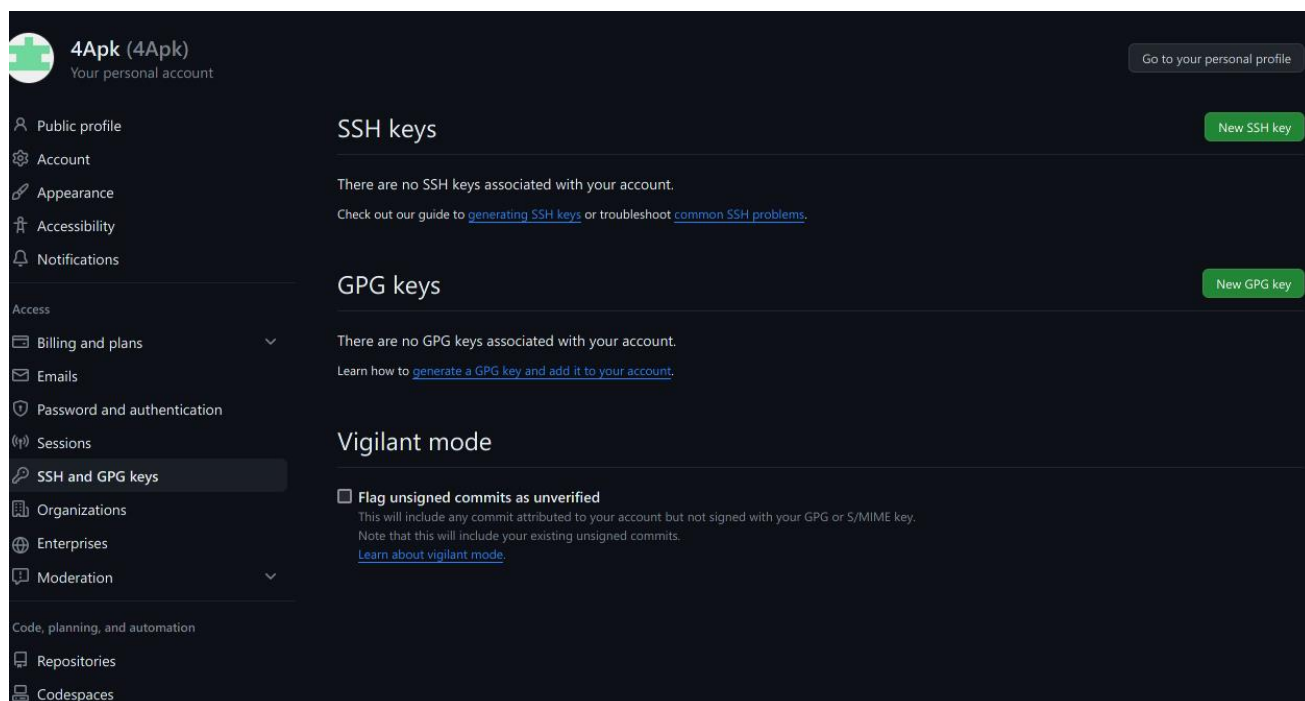


Рис. 4.9: Окно SSH and GPG keys

Вставляю скопированный ключ в поле «Key». В поле Title указываю имя для ключа. Нажимаю «Add SSH-key», чтобы завершить добавление ключа (рис. 4.10).

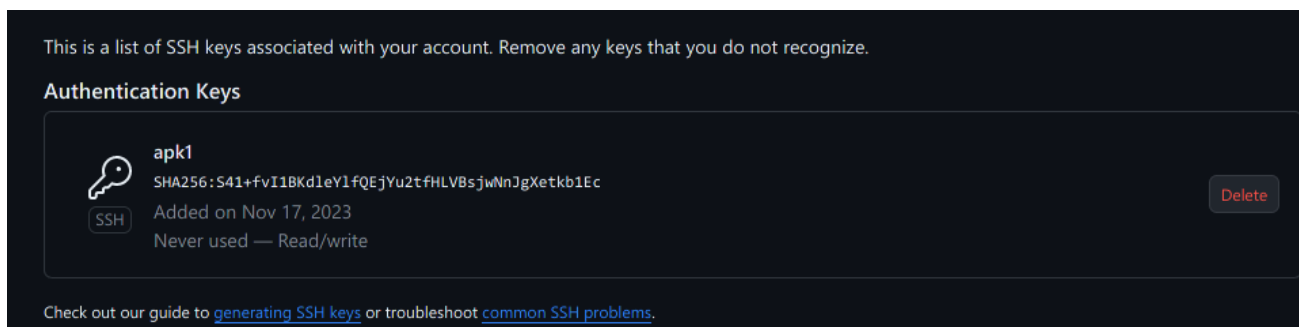


Рис. 4.10: Добавил ключ

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. 4.11).

```
apkorchagin@dk3n64 ~ $ ls
Kol.txt  public  public_html  work  Видео  Документы  Загрузки  Изображения  Музыка  Общедоступные  'Рабочий стол'
```

Рис. 4.11: 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. 4.12).

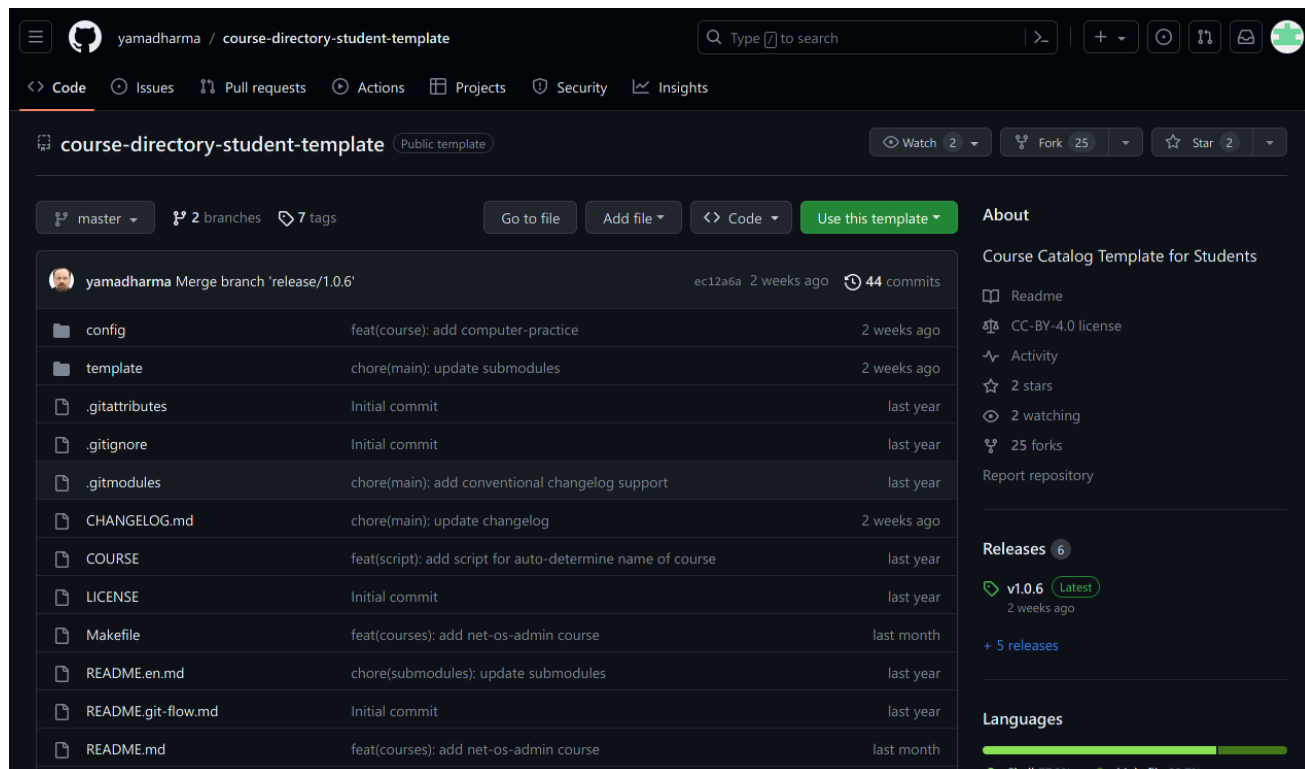


Рис. 4.12: Страница шаблона для репозитория

В открывшемся окне задаю имя репозитория (Repository name): study_2023–2024_arh-rc и создаю репозиторий, нажимаю на кнопку «Create repository from template» (рис. 4.13).

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 * 4Apk / Repository name * study_2023-2024_arh-pc

✔ study_2023-2024_arh-pc is available.

Great repository names are short and memorable. Need inspiration? How about [psychic-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. 4.13: Окно создания репозитория

Репозиторий создан (рис. 4.14).

study_2023-2024_arh-pc Public Pin Unwatch 1

generated from [yamadharm/course-directory-student-template](#)

master 1 branch 0 tags Go to file Add file Code

4Apk Initial commit ae72f2b 1 minute ago 1 commit

config	Initial commit	1 minute ago
template	Initial commit	1 minute ago
.gitattributes	Initial commit	1 minute ago
.gitignore	Initial commit	1 minute ago
.gitmodules	Initial commit	1 minute ago
CHANGELOG.md	Initial commit	1 minute ago
COURSE	Initial commit	1 minute ago
LICENSE	Initial commit	1 minute ago
Makefile	Initial commit	1 minute ago
README.en.md	Initial commit	1 minute ago
README.git-flow.md	Initial commit	1 minute ago
README.md	Initial commit	1 minute ago

Рис. 4.14: Созданный репозиторий

Через терминал перехожу в созданный каталог курса с помощью утилиты `cd` (рис. 4.15).

```
apkorchagin@dk3n64 ~ $ cd ~/work/study/2023-2024/"Архитектура компьютера"
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера $
```

Рис. 4.15: Перемещение между директориями

Клонирую созданный репозиторий с помощью команды `git clone --recursive git@github.com:/study_2023-2024_arh-pc.git arch-pc` (рис. 4.16).

```
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера $ git clone --recursive git@github.com:4Apk/study_2023-2024_arh-pc.git arch-pc
Клонирование в «arch-pc»...
The authenticity of host 'github.com (140.82.121.4)' 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.
remote: Enumerating objects: 30, done.
remote: Counting objects: 100% (30/30), done.
remote: Compressing objects: 100% (29/29), done.
remote: Total 30 (delta 1), reused 17 (delta 0), pack-reused 0
Получение объектов: 100% (30/30), 17.75 КБ | 4.44 МБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/afs/.dk.sci.pfu.edu.ru/home/a/p/apkorchagin/work/study/2023-2024/Архитектура компьютера/arch-pc/template/presentation»...
remote: Enumerating objects: 95, done.
remote: Counting objects: 100% (95/95), done.
remote: Compressing objects: 100% (67/67), done.
remote: Total 95 (delta 34), reused 87 (delta 26), pack-reused 0
Получение объектов: 100% (95/95), 96.99 КБ | 1.05 МБ/с, готово.
Определение изменений: 100% (34/34), готово.
Клонирование в «/afs/.dk.sci.pfu.edu.ru/home/a/p/apkorchagin/work/study/2023-2024/Архитектура компьютера/arch-pc/template/report»...
remote: Enumerating objects: 112, done.
remote: Counting objects: 100% (112/112), done.
remote: Compressing objects: 100% (77/77), done.
remote: Total 112 (delta 45), reused 98 (delta 31), pack-reused 0
Получение объектов: 100% (112/112), 331.19 КБ | 2.42 МБ/с, готово.
Определение изменений: 100% (45/45), готово.
Submodule path 'template/presentation': checked out '40a1761813e197d00e8443ff1ca72c60a304f24c'
Submodule path 'template/report': checked out '25e169d367953f60c76c251db299ed52852b401f'
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера $
```

Рис. 4.16: Клонирование репозитория

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку “SSH” (рис 4.17)

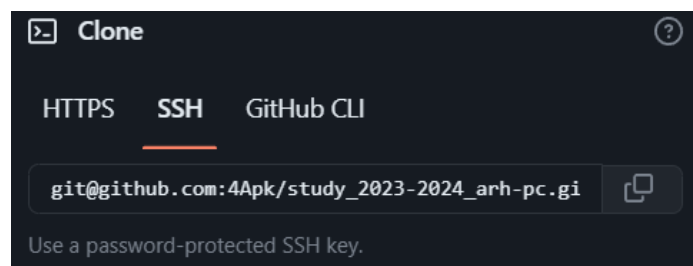


Рис. 4.17: Ссылка на клонирование

4.6 Настройка каталога курса

Перехожу в каталог `arch-pc` с помощью утилиты `cd` и удаляю лишние файлы при

помощи gm (рис. 4.18) .


```

apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера $ cd ~/work/study/2023-2024/Архитектура компьютера/arch-pc
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ rm package.json
bash: rm.package.json: команда не найдена
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ rm package.json

```

Рис. 4.18: Удаление файлов

Создаю необходимые каталоги (рис. 4.19).

```

apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ echo arch-pc > COURSE
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ make
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ 

```

Рис. 4.19: 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. 4.20).

```

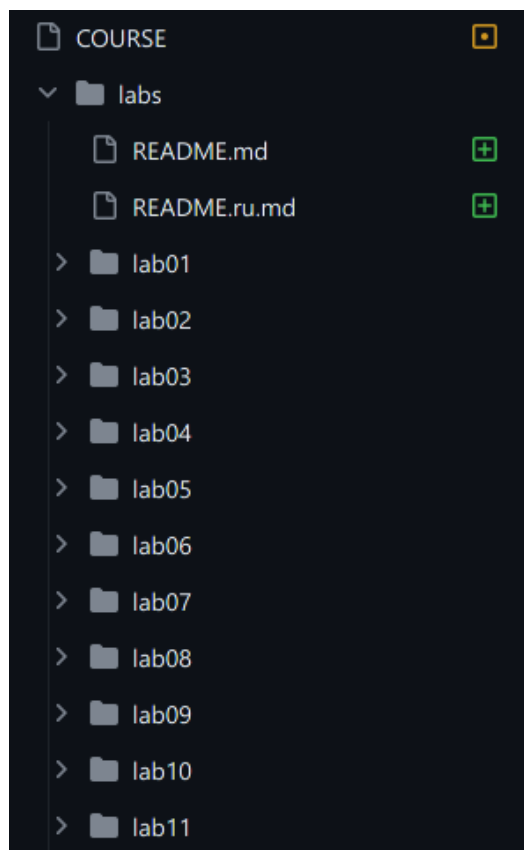
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ git add .
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ git commit -am 'feat(main): make course structure'
[master 455fcb7] feat(main): make course structure
199 files changed, 54725 insertions(+), 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

```

Отправляю все на сервер с помощью push (рис. 4.21).

```
аркoгсhаgіn@к3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc $ git push
Перечисление объектов: 37, готово.
Подсчет объектов: 100% (37/37), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (29/29), готово.
Запись объектов: 100% (35/35), 342.14 КиБ | 9.00 МиБ/с, готово.
Всего 35 (изменений 4), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:4Apk/study_2023-2024_arh-pc.git
   ae72f2b..455fcb7  master -> master
```

Проверяю правильность выполнения работы сначала на самом сайте GitHub (рис. 4.22).



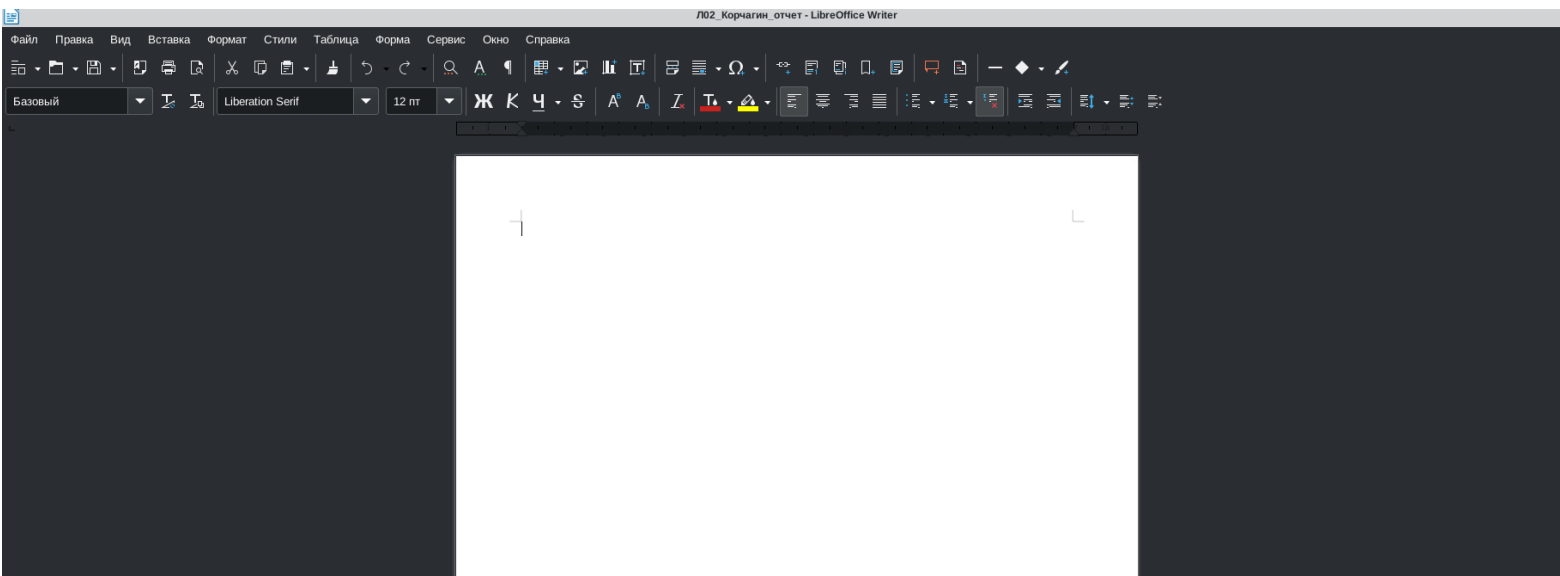
4.7 Выполнение заданий для самостоятельной работы

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. 4.23).

```
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab02/report $ touch Л02_Корчагин_отчет
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab02/report $
```

Рис. 4.23: Создание файла

Оформить отчет я смогу в текстовом процессоре LibreOffice Writer, найдя его в меню приложений (рис. 4.24).



2. Перехожу из подкаталога lab02/report в подкаталог lab01/report с помощью утилиты cd (рис. 4.25).

```
apkorchagin@dk3n64 ~ $ cd ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc/labs/lab01/report/  
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report $
```

Проверяю местонахождение файла с отчетом по первой лабораторной работе. Он должен быть в подкаталоге домашней директории «Загрузки», для проверки использую команду ls (рис. 4.26).

```
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report $ ls ~/Загрузки  
'отчет_лаб_1, Корчагин А.П. НММ-02-23.doc'
```

Копирую первую лабораторную с помощью утилиты cp и проверяю правильность выполнения команды cp с помощью ls (рис. 4.27).

```
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report $ ls  
bib image Makefile pandoc report.md Л01_Корчагин_отчет.doc
```

3. Добавляю с помощью команды `git add` в коммит созданные файлы:
Л01_Корчагин_отчет (рис. 4.28).

```
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report $ git add Л01_Корчагин_отчет.doc
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report $
```

Сохраняю изменения на сервере командой `git commit -m "..."`, поясняя, что добавила файлы(4.29).

```
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report $ git commit -m "Add existing Files"
[master 63ea025] Add existing Files
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100755 labs/lab01/report/Л01_Корчагин_отчет.doc
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report $
```

То же самое делаю для отчета по второй лабораторной работе: перехожу в директорию `labs/lab02/report` с помощью `cd`, добавляю с помощью `git add` нужный файл, сохраняю изменения с помощью `git commit` (рис. 4.30).

```
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab02/report $ git add Л02_Корчагин_отчет.doc
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab02/report $ git commit -m "add exsting file"
[master 5ab8c01] add exsting file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 labs/lab02/report/Л02_Корчагин_отчет.doc
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab02/report $
```

Отправляю в центральный репозиторий сохраненные изменения командой `git push -f origin master` (рис. 4.31).

```
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab02/report $ git push -f origin master
Перечисление объектов: 17, готово.
Подсчет объектов: 100% (15/15), готово.
При сжатии изменений используется до 6 потоков
Сжатие объектов: 100% (11/11), готово.
Запись объектов: 100% (11/11), 525.09 КиБ | 1.91 МиБ/с, готово.
Всего 11 (изменений 5), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (5/5), completed with 2 local objects.
To github.com:4Apk/study_2023-2024_arh-pc.git
  455fcb7..5ab8c01  master -> master
apkorchagin@dk3n64 ~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab02/report $
```

При просмотре изменений на Github вижу, что были добавлены файлы с отчетами по лабораторным работам (рис. 4.32-4.33) .

4Apk Add existing Files 63ea025	
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
ЛО1_Корчагин_отчет.doc	Add existing Files

4Apk add exsting file 5ab8c01	
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
Л02_Корчагин_отчет.doc	add exsting file

Рис. 4.33: 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.

