

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 2

дисциплина: Архитектура компьютера

Студент: Горичев Д.В

Группа: НКАбд–02-23

МОСКВА

2023 г.

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	10
4.5	Создание репозитория курса на основе шаблона.....	10
4.6	Настройка каталога курса	13
4.7	Выполнение заданий для самостоятельной работы.....	16
5	Выводы	20

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.

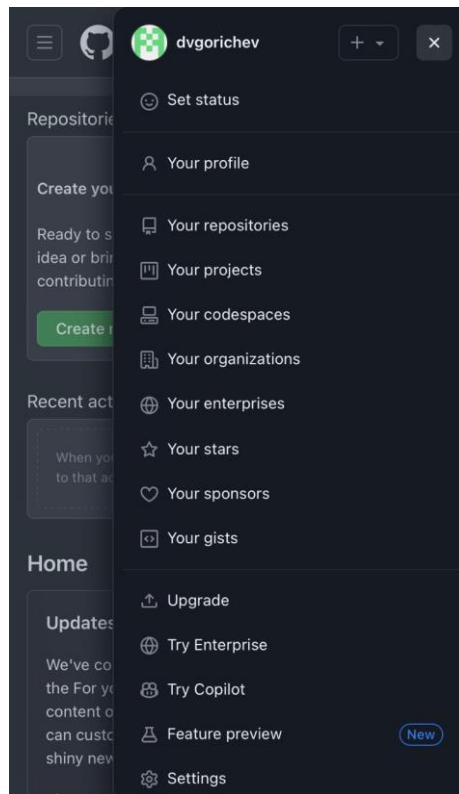


Рис. 1 - Аккаунт GitHub

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.

```
dvgorichev@dvgorichev:~$ git config --global user.name "<Daniil Gorichev>"
dvgorichev@dvgorichev:~$ git config --global user.email "<1132239967@pfur.ru>"
```

Рис. 2 - Предварительная конфигурация git

Настраиваю utf-8 в выводе сообщений git для корректного отображения символов.

```
dvgorichev@dvgorichev:~$ git config --global core.quotePath false
```

Рис. 3 - Настройка кодировки

Задаю имя «master» для начальной ветки.

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

Рис. 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. CR и LF – это символы, которые можно использовать для обозначения разрыва строки в текстовых файлах.

```
dvgorichev@dvgorichev:~$ git config --global core.autocrlf input
```

Рис. 5 - Параметр autocrlf

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

```
dvgorichev@dvgorichev:~$ git config --global core.safecrlf warn
```

Рис. 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. Ключ автоматически сохранится в каталоге `~/.ssh/`.


```

dvgorichev@dvgorichev:~$ ssh-keygen -C "dvgorichev <1132239967@pfur.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/dvgorichev/.ssh/id_rsa):
Created directory '/home/dvgorichev/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/dvgorichev/.ssh/id_rsa
Your public key has been saved in /home/dvgorichev/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:LbaWHJaJ+YQuM30IU8Q/rH2Cq0dZurp2kZhbgJJAIrw dvgorichev <1132239967@pfur.ru>
The key's randomart image is:
+---[RSA 3072]-----+
|=.
|+. .
|... o
|oE . o + +
|. = o B S .
| + B * B =
| * @ = B
| +. * * +
|==*.
+-----[SHA256]-----+

```

Рис. 7 - Генерация SSH-ключа

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

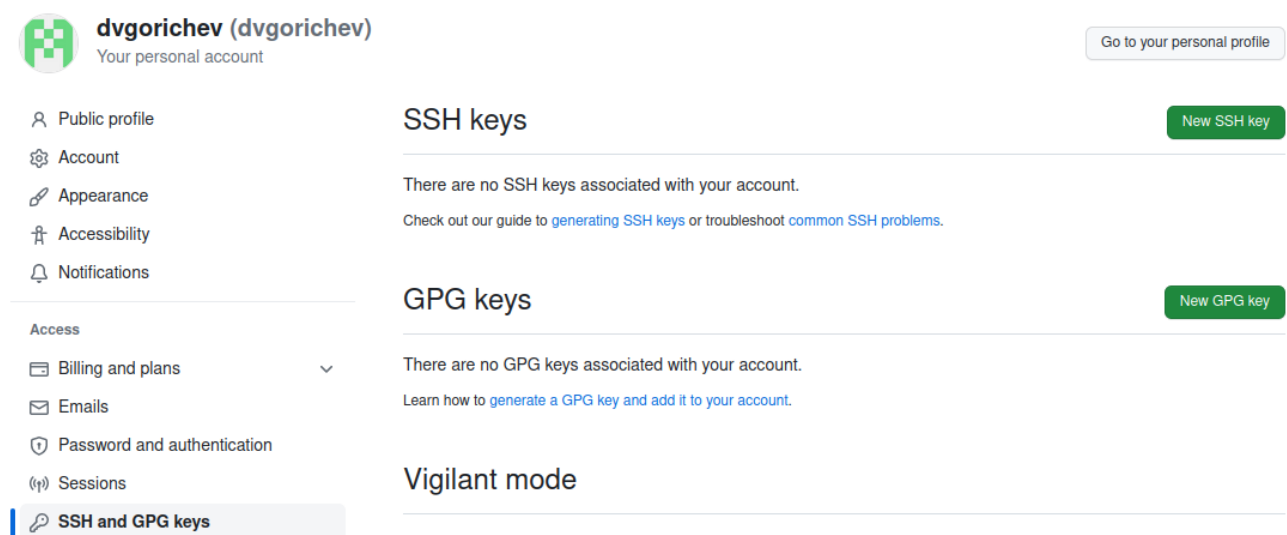


Рис. 8 - Окно SSH and GPG keys

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

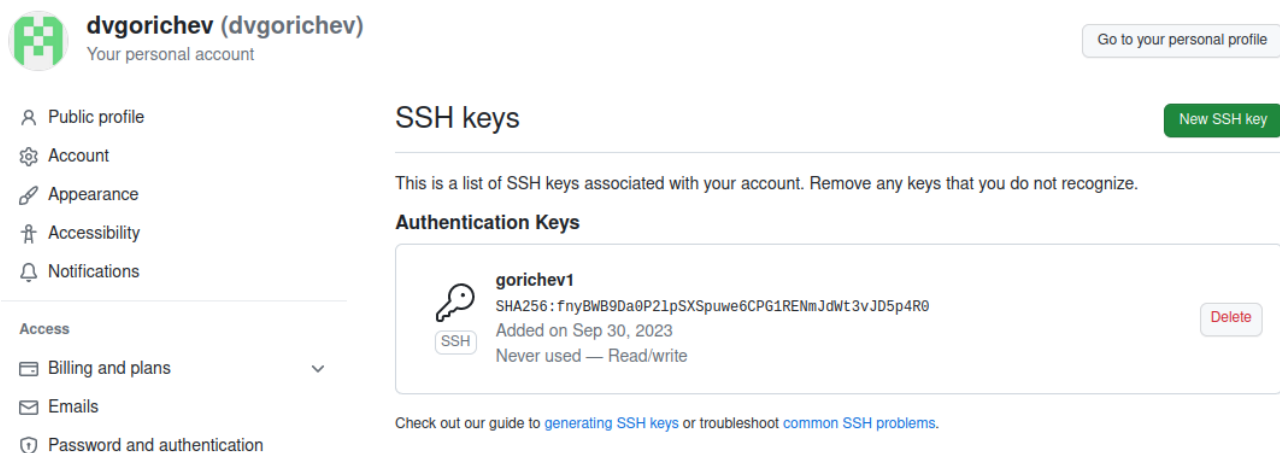


Рис. 9 - Добавление ключа

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.

```
dvgorichev@dvgorichev:~$ mkdir -p work/study/2023-2024/"Архитектура компьютера"
dvgorichev@dvgorichev:~$ ls
dvgorichev  work  Документы  Изображения  Общедоступные  Шаблоны
snap        Видео  Загрузки   Музыка       'Рабочий стол'
```

Рис. 10 - 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.


master

Go to file

Add file

<> Code

Use this template

 **yamadharm** Merge branch 'release/1.0.4'


c6ab3e6 on Feb 7 34 commits

config	feat(course): add presentation element	7 months ago
template	chore(template): update templates	7 months ago
.gitattributes	Initial commit	last year
.gitignore	Initial commit	last year
.gitmodules	chore(main): add conventional changelog support	last year
CHANGELOG.md	chore(main): update changelog	7 months ago
COURSE	feat(script): add script for auto-determine name of c...	last year
LICENSE	Initial commit	last year

Рис. 11 - Страница шаблона для репозитория

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


Repository template

 yamadharm/course-directory-student-template

Start your repository with a template repository's contents.


☐ **Include all branches**
Copy all branches from yamadharm/course-directory-student-template and not just the default branch.

Owner *

 dvgorichev


Repository name *


study_2023-2024_arh-pc


 The repository study_2023-2024_arh-pc already exists on this account.

Great repository names are short and memorable. Need inspiration? How about **effective-goggles** ?

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. 12 - Окно создания репозитория

Репозиторий создан.

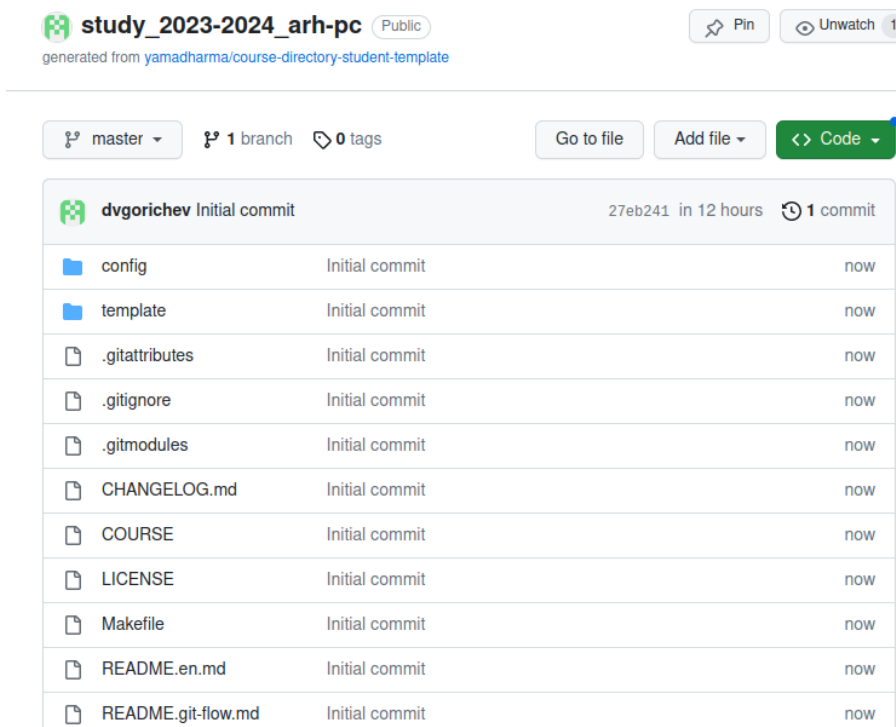


Рис. 13 - Созданный репозиторий

Через терминал перехожу в созданный каталог курса с помощью утилиты cd.

```
dvgorichev@dvgorichev:~$ cd work/study/2023-2024/'Архитектура компьютера'  
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера$
```

Рис. 14 - Перемещение между директориями

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

```

dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера$ git clone --recursive git@github
b.com:dvgorichev/study_2023-2024_arh-pc.git arch-pc
Клонирование в «arch-pc»...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 27 (delta 1), reused 11 (delta 0), pack-reused 0
Получение объектов: 100% (27/27), 16.93 КиБ | 8.47 МиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharm/academic-presentation-markdown-temp
late.git) зарегистрирован по пути «template/presentation»
Подмодуль «template/report» (https://github.com/yamadharm/academic-laboratory-report-template.git)
зарегистрирован по пути «template/report»
Клонирование в «/home/dvgorichev/work/study/2023-2024/Архитектура компьютера/arch-pc/template/presen
tation»...
remote: Enumerating objects: 82, done.
remote: Counting objects: 100% (82/82), done.
remote: Compressing objects: 100% (57/57), done.
remote: Total 82 (delta 28), reused 77 (delta 23), pack-reused 0
Получение объектов: 100% (82/82), 92.90 КиБ | 60.00 КиБ/с, готово.
Определение изменений: 100% (28/28), готово.
Клонирование в «/home/dvgorichev/work/study/2023-2024/Архитектура компьютера/arch-pc/template/report

```

Рис. 15 - Клонирование репозитория

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

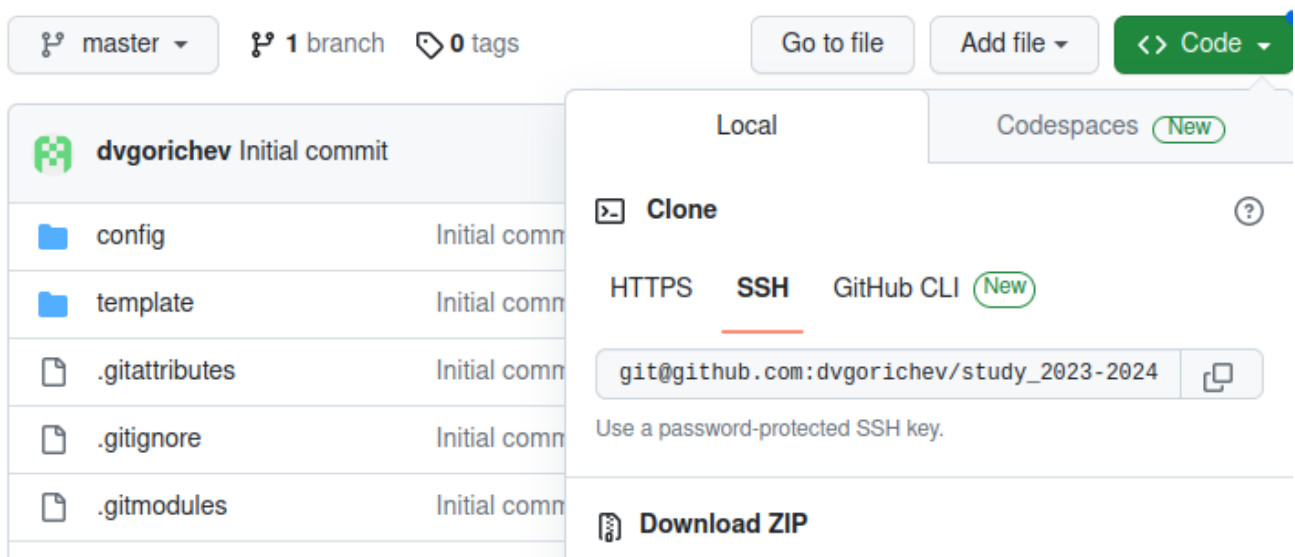


Рис. 16 - Окно с ссылкой для копирования репозитория

4.6 Настройка каталога курса

Перехожу в каталог arch-pc с помощью утилиты cd.

```
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера$ cd arch-pc  
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc$
```

Рис. 17 - Перемещение между директориями

Удаляю лишние файлы с помощью утилиты rm.

```
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ rm package.json
```

Рис. 18 - Удаление файлов

Создаю необходимые каталоги.

```
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ echo arch-pc > COURSE  
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ make
```

```
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ make  
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ ls  
CHANGELOG.md  COURSE  LICENSE  prepare  README.en.md  README.md  
config        labs    Makefile  presentation  README.git-flow.md  template
```

Рис. 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.

```

dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ git add .
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc$ git commit -am 'feat(main): make course structure'
[master f653330] feat(main): make course structure
199 files changed, 54725 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
create mode 100644 labs/lab01/presentation/Makefile
create mode 100644 labs/lab01/presentation/image/kulyabov.jpg
create mode 100644 labs/lab01/presentation/presentation.md
create mode 100644 labs/lab01/report/Makefile
create mode 100644 labs/lab01/report/bib/cite.bib
create mode 100644 labs/lab01/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab01/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_fignos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_secnos.py
create mode 100755 labs/lab01/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/core.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/main.py
create mode 100644 labs/lab01/report/pandoc/filters/pandocxnos/pandocattributes.py
create mode 100644 labs/lab01/report/report.md
create mode 100644 labs/lab02/presentation/Makefile
create mode 100644 labs/lab02/presentation/image/kulyabov.jpg
create mode 100644 labs/lab02/presentation/presentation.md
create mode 100644 labs/lab02/report/Makefile
create mode 100644 labs/lab02/report/bib/cite.bib
create mode 100644 labs/lab02/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab02/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_eqnos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_fignos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_secnos.py
create mode 100755 labs/lab02/report/pandoc/filters/pandoc_tablenos.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/__init__.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/core.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/main.py
create mode 100644 labs/lab02/report/pandoc/filters/pandocxnos/pandocattributes.py

```

Рис. 20 - Добавление и сохранение изменений на сервере

Отправляю все на сервер с помощью push.

```

dvgorichev@dvgorichev:~/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 3 потоков
Сжатие объектов: 100% (29/29), готово.
Запись объектов: 100% (35/35), 342.13 КиБ | 370.00 КиБ/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:dvgorichev/study_2023-2024_arh-pc.git
 27eb241..f653330 master -> master

```

Рис. 21 - Выгрузка изменений на сервер

Проверяю правильность выполнения работы сначала на самом сайте GitHub.

study_2023-2024_arh-pc / labs /		↑ Top
lab02	feat(main): make course structure	2 minutes ago
lab03	feat(main): make course structure	2 minutes ago
lab04	feat(main): make course structure	2 minutes ago
lab05	feat(main): make course structure	2 minutes ago
lab06	feat(main): make course structure	2 minutes ago
lab07	feat(main): make course structure	2 minutes ago
lab08	feat(main): make course structure	2 minutes ago
lab09	feat(main): make course structure	2 minutes ago
lab10	feat(main): make course structure	2 minutes ago
lab11	feat(main): make course structure	2 minutes ago
README.md	feat(main): make course structure	2 minutes ago
README.ru.md	feat(main): make course structure	2 minutes ago

Рис. 22 - Страница репозитория

4.7 Выполнение заданий для самостоятельной работы

1. Перехожу в директорию labs/lab02/report с помощью утилиты cd. Создаю в каталоге файл для отчета по третьей лабораторной работе с помощью утилиты touch.

```
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab02/report$ touch Л02_Горичев.отчет
```

Рис. 23 - Создание файла

2. Перехожу из подкаталога lab02/report в подкаталог lab01/report с помощью утилиты cd.

```
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab02$ cd ..
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs$ cd lab01
```

Рис. 24 - Перемещение между директориями

Проверяю местонахождение файлов с отчетами по первой лабораторной работе. Они должны быть в подкаталоге домашней директории «Загрузки», для проверки использую команду ls.

```
dvgorichev@dvgorichev:/home$ ls ~/Загрузки  
'Telegram Desktop' Л01_Горичев_Отчет.pdf
```

Рис. 25 - Проверка местонахождения файлов

Копирую первую лабораторную с помощью утилиты cp и проверяю правильность выполнения команды cp с помощью ls.

```
dvgorichev@dvgorichev:~/work/study/2023-2024$ cp ~/Загрузки/Л01_Горичев_Отчет.pdf /home/dvgorichev/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report  
  
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report$ ls  
bib image Makefile pandoc report.md Л01_Горичев_Отчет.pdf
```

Рис. 26 - Копирование и проверка файла

Аналогичным образом сделаем данную работу и со вторым отчетом.

3. Добавляю с помощью команды git add в коммит созданные файлы: Л01_Горичев отчет. Точно таким же методом сделаем работу со вторым отчетом.

```
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report$ git add Л01_Горичев_Отчет.pdf
```

Рис. 27 - Добавление файла на сервер


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


Отправляю в центральный репозиторий сохраненные изменения командой git push -f origin master.

```
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report$ git commit -m "Л01_Горичев_Отчет.pdf"
[master 6153281] Л01_Горичев_Отчет.pdf
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 labs/lab01/report/Л01_Горичев_Отчет.pdf
dvgorichev@dvgorichev:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab01/report$ git push -f origin master
Перечисление объектов: 10, готово.
Подсчет объектов: 100% (10/10), готово.
При сжатии изменений используется до 3 потоков
Сжатие объектов: 100% (6/6), готово.
Запись объектов: 100% (6/6), 312.72 КиБ | 2.19 МиБ/с, готово.
Всего 6 (изменений 3), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To github.com:dvgorichev/study_2023-2024_arh-pc.git
 f653330..6153281 master -> master
```

Рис. 28 - Отправка в центральный репозиторий сохраненных изменений

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

 dvgorichev Л01_Горичев_Отчет.pdf

6153281 · 1 minute ago  History

Name	Last commit message	Last commit date
..		
bib	feat(main): make course structure	4 hours ago
image	feat(main): make course structure	4 hours ago
pandoc	feat(main): make course structure	4 hours ago
Makefile	feat(main): make course structure	4 hours ago
report.md	feat(main): make course structure	4 hours ago
Л01_Горичев_Отчет.pdf	Л01_Горичев_Отчет.pdf	1 minute ago

Рис. 29 - Страница каталога в репозитории

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

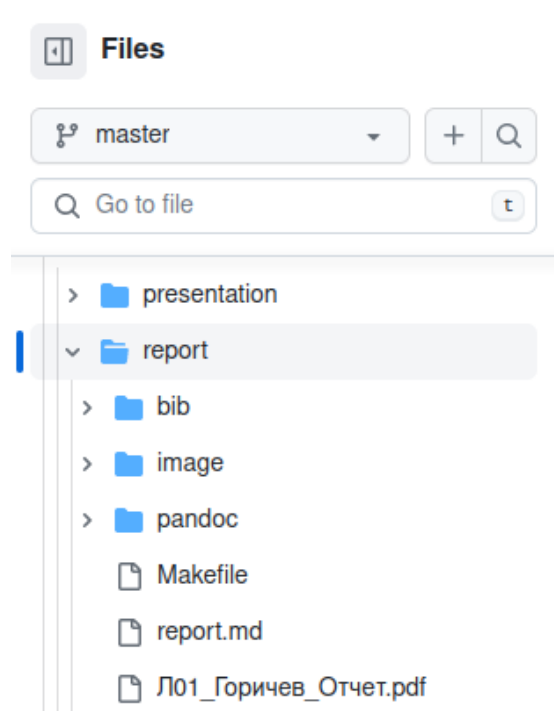


Рис. 30 - Страница последних изменений в репозитории

5 Выводы

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