

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: Заверняев А. И.

Группа: НКАбд-03-23

№ ст. билета: 1032230326

МОСКВА

2023 г.

Содержание

1	Цель работы	3
2	Задание	4
3	Теоретическое введение	5
4	Выполнение лабораторной работы	6
4.1	Настройка GitHub	6
4.2	Базовая настройка Git	6
4.3	Создание SSH-ключа	7
4.4	Создание рабочего пространства	8
4.5	Настройка каталога курса	11
4.6	Выполнение заданий для самостоятельной работы	13
5	Выводы	15
6	Список литературы	16

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.

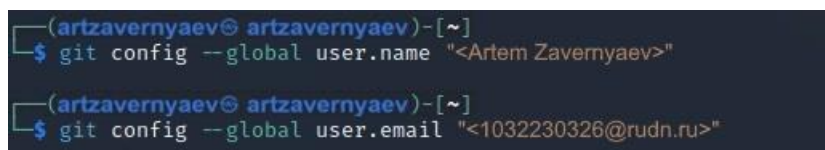
4 Выполнение лабораторной работы

4.1 Настройка GitHub

Так как у меня уже есть учетная запись на сайте GitHub, я пропущу процесс её создания.

4.2 Базовая настройка Git

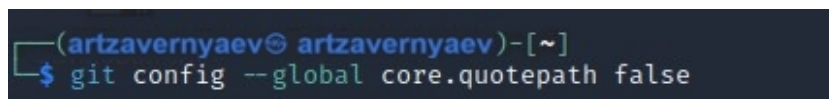
Открываю виртуальную машину, затем открываю терминал и делаю предварительную конфигурацию git. Ввожу команду `git config --global user.name ""`, указывая свое имя и команду `git config --global user.email "work@mail"`, указывая в ней свою электронную почту (корпоративную почту) (рис. 4.1).



```
(artzavernyaev@ artzavernyaev)-[~]  
$ git config --global user.name "<Artem Zavernyaev>"  
  
(artzavernyaev@ artzavernyaev)-[~]  
$ git config --global user.email "<1032230326@rudn.ru>"
```

Рис. 4.1: Предварительная конфигурация git

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



```
(artzavernyaev@ artzavernyaev)-[~]  
$ git config --global core.quotepath false
```

Рис. 4.2: Настройка кодировки

Задаю имя «master» для начальной ветки (рис. 4.3).

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

Рис. 4.3: 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. 4.4). CR и LF – это символы, которые можно использовать для обозначения разрыва строки в текстовых файлах.

```
(artzavernyaev@ artzavernyaev)-[~]
$ git config --global core.autocrlf input
```

Рис. 4.4: Параметр `autocrlf`

Задаю параметр `safecrlf` со значением `warn`, так Git будет проверять преобразование на обратимость (рис. 4.5).

```
(artzavernyaev@ artzavernyaev)-[~]
$ git config --global core.safecrlf warn
```

Рис. 4.5: 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 своё имя и свою электронную почту (рис. 4.6). Ключ автоматически сохранится в каталоге `~/.ssh/`.

Рис. 4.6: Генерация SSH-ключа

```
(artzavernyaev@ artzavernyaev)-[~]
$ ssh-keygen -C "Artem Zavernyaev <1032230326@rudn.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/artzavernyaev/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/artzavernyaev/.ssh/id_rsa
Your public key has been saved in /home/artzavernyaev/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:ovXv5WrLegR/WXpJswarONgS6FRLM7Q4Mi+gPWaYM2s Artem Zavernyaev <1032230326@rudn.ru>
The key's randomart image is:
+----[RSA 3072]-----+
|
|  o . o . . .
| * . = o +
| o B = oo+
| = B . =*
| S * . .o
| . + +.
| . . . .+
| . . . .o
| . . . .o
| . . . .o
+-----[SHA256]-----+
```

Копирую открытый ключ из директории, в которой он был сохранен.

Открываю браузер, захожу на сайт GitHub. Открываю свой профиль и выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key» (рис. 4.7).

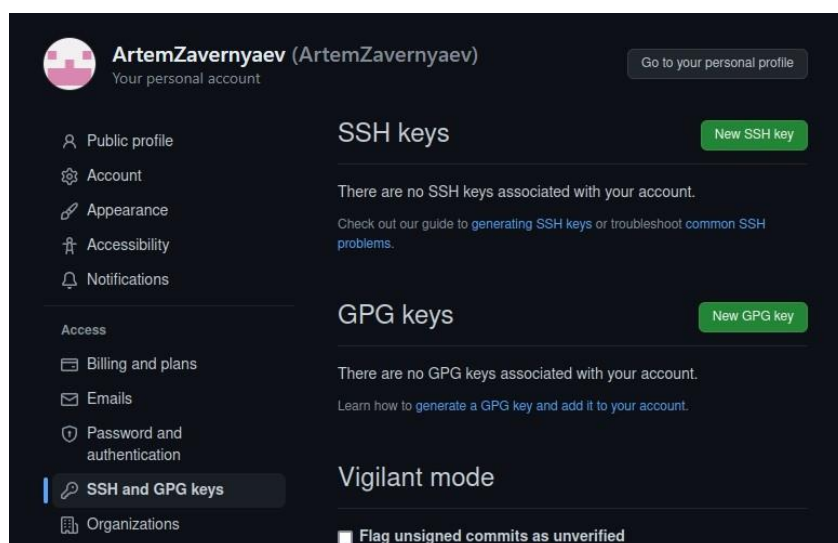


Рис. 4.7: Окно SSH and GPG keys

Вставляю скопированный ключ в поле «Key». В поле Title указываю имя для ключа. Нажимаю «Add SSH-key», чтобы завершить добавление ключа (рис. 4.8).

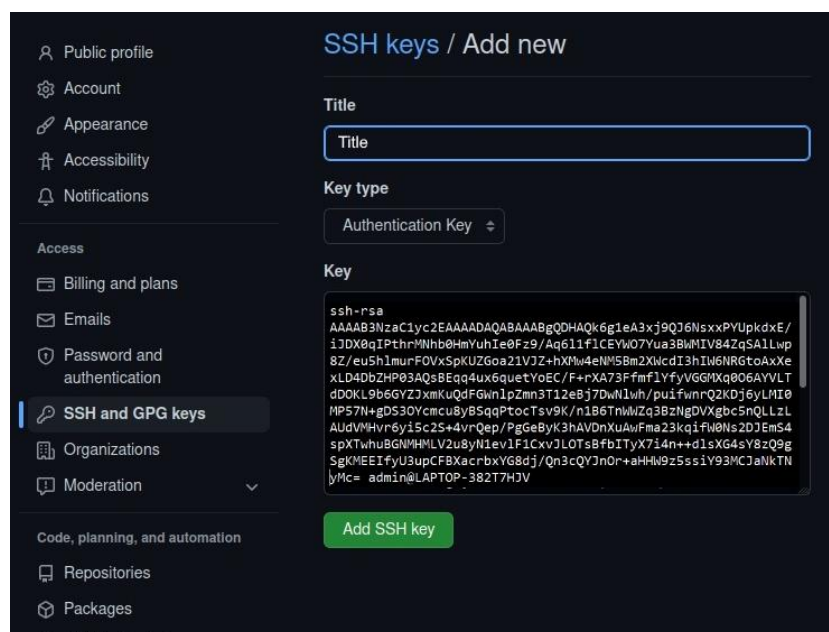


Рис. 4.8: Добавление ключа

4.4 Создание рабочего пространства

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. 4.9).

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

(artzavernyaev@ artzavernyaev)~$ ls
install      work      Документы  Изображения  Общедоступные  Шаблоны
parentdir    Видео     Загрузки   Музыка        'Рабочий стол'
```

Рис. 4.9: 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. 4.10).

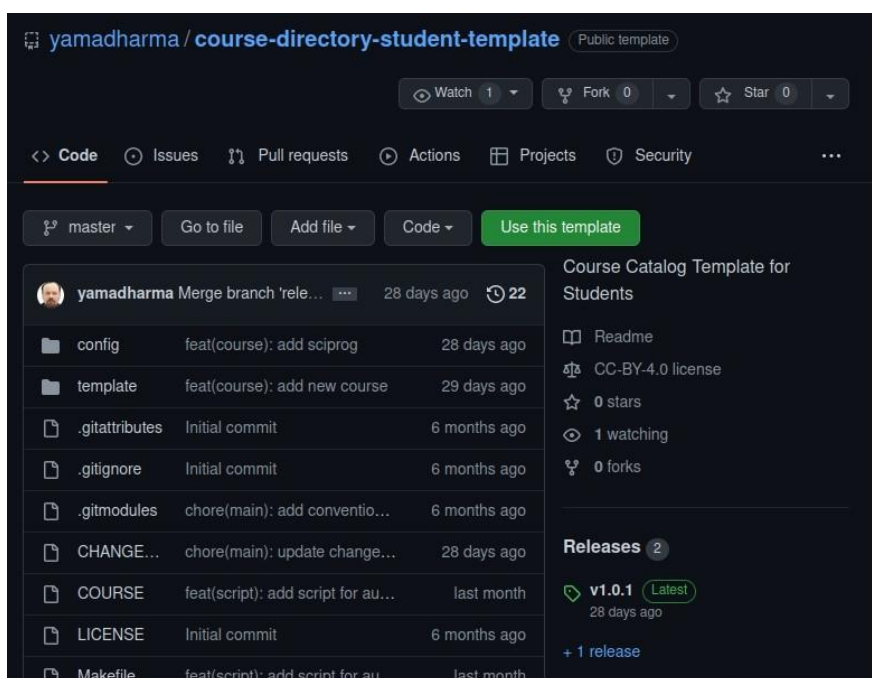


Рис. 4.10: Страница шаблона для репозитория

В открывшемся окне задаю имя репозитория (Repository name): `study_2022–2023_arh-` и создаю репозиторий, нажимая на кнопку «Create repository from template» (рис. 4.11).

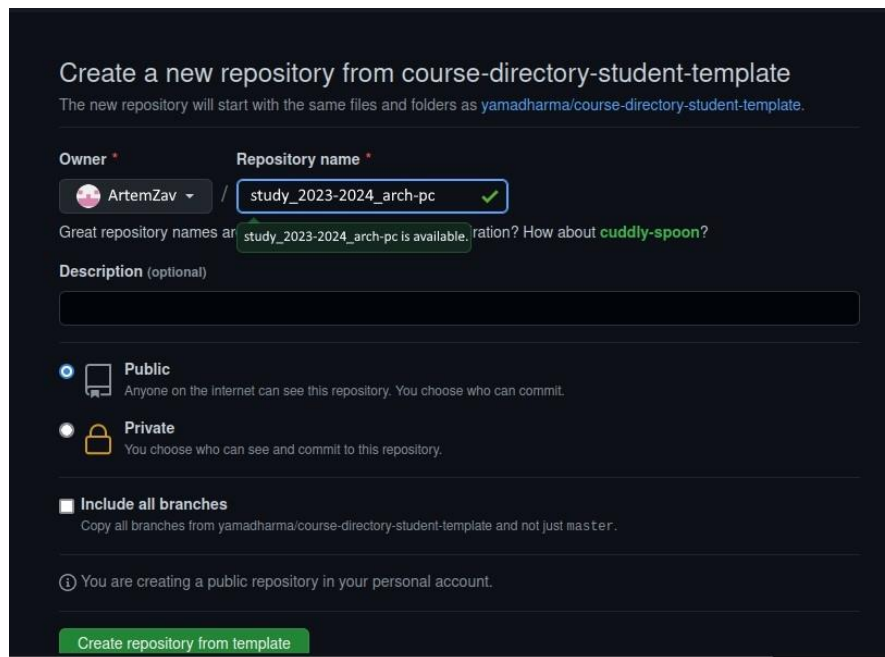


Рис. 4.11: Окно создания репозитория

Проверю, что репозиторий создан (рис. 4.12).

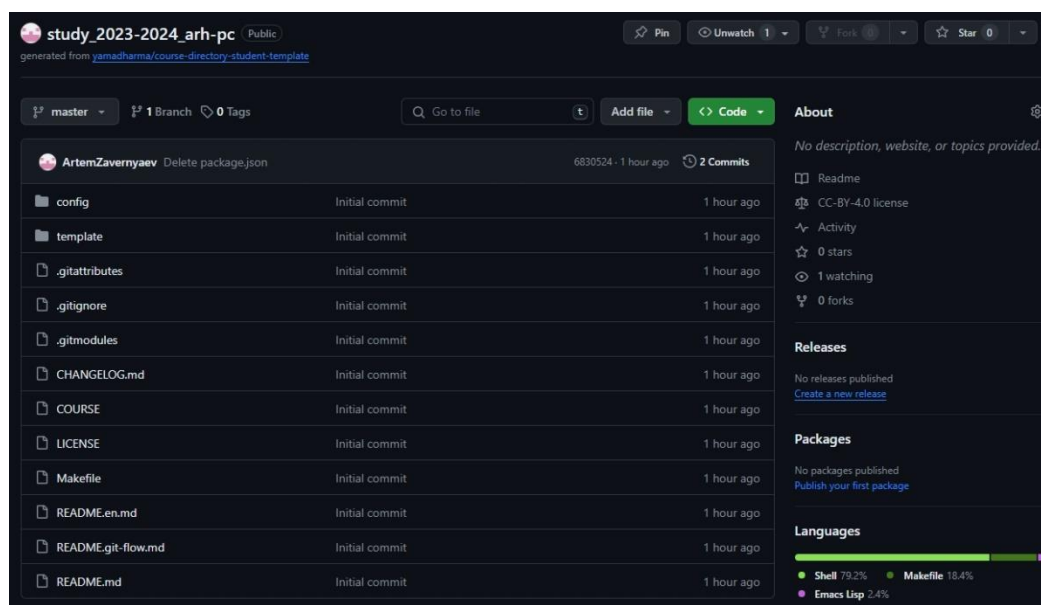


Рис. 4.12: Созданный репозиторий

Через терминал перехожу в созданный каталог курса с помощью утилиты cd (рис. 4.13).

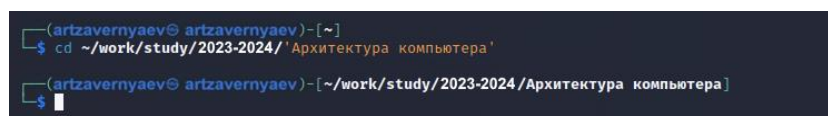


Рис. 4.13: Перемещение между директориями

Клонирую созданный репозиторий с помощью команды `git clone --recursive git@github.com:/study_2023-2024_arh-pc.git arch-pc` (рис. 4.14).

```
(artzavernyaev@ artzavernyaev) - [~/work/study/2023-2024/Архитектура компьютера]
$ git clone --recursive git@github.com:ArtemZavernyaev/study_2023-2024_arh-pc.git arch-pc
Клонирование в «arch-pc»...
The authenticity of host 'github.com (140.82.121.3)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6Tul1hhoZisF/zlDA0zPMSvHdKkr4UvC0qU.
```

Рис. 4.14: Клонирование репозитория

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH» (рис. 4.15).

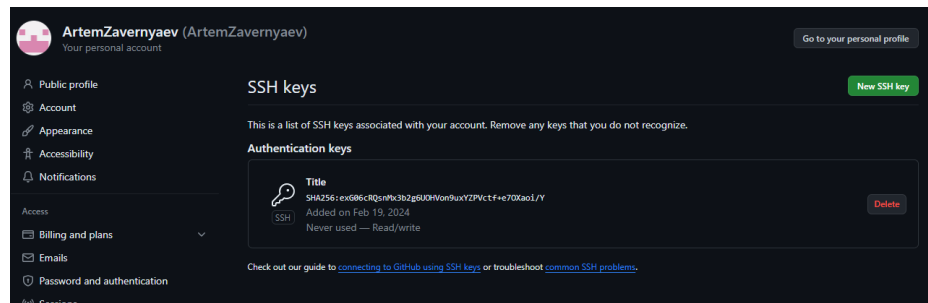


Рис. 4.15: Окно с ссылкой для копирования репозитория

4.5 Настройка каталога курса

Перехожу в каталог arch-pc с помощью утилиты cd (рис. 4.16).

```
(artzavernyaev@ artzavernyaev) - [~/work/study/2023-2024/Архитектура компьютера]
$ cd ~/work/study/2023-2024/Архитектура\ компьютера/arch-pc
(artzavernyaev@ artzavernyaev) - [~/.../study/2023-2024/Архитектура компьютера/arch-pc]
$
```

Рис. 4.16: Перемещение между директориями

Удаляю лишние файлы с помощью утилиты rm (рис. 4.17).

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

Рис. 4.17: Удаление файлов

Создаю необходимые каталоги (рис. 4.18).

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

Рис. 4.18: 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. 4.19).

```

(artzavernyaev@ artzavernyaev)-[~/../study/2023-2024 /Архитектура компьютера/arch-pc]
$ git add .

(artzavernyaev@ artzavernyaev)-[~/../study/2023-2024 /Архитектура компьютера/arch-pc]
$ git commit -am 'feat(main): make course structure'
[master 2f55d30] feat(main): make course structure
91 files changed, 8229 insertions(+), 14 deletions(-)
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 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 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
create mode 100644 labs/lab03/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab03/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab03/report/report.md
create mode 100644 labs/lab04/presentation/Makefile
create mode 100644 labs/lab04/presentation/image/kulyabov.jpg
create mode 100644 labs/lab04/presentation/presentation.md
create mode 100644 labs/lab04/report/Makefile
create mode 100644 labs/lab04/report/bib/cite.bib
create mode 100644 labs/lab04/report/image/placeimg_800_600_tech.jpg
create mode 100644 labs/lab04/report/pandoc/csl/gost-r-7-0-5-2008-numeric.csl
create mode 100644 labs/lab04/report/report.md
create mode 100644 labs/lab05/presentation/Makefile
create mode 100644 labs/lab05/presentation/image/kulyabov.jpg
create mode 100644 labs/lab05/presentation/presentation.md
create mode 100644 labs/lab05/report/Makefile
create mode 100644 labs/lab05/report/bib/cite.bib
create mode 100644 labs/lab05/report/image/placeimg_800_600_tech.jpg

```

Рис. 4.19: Добавление и сохранение изменений на сервере

Отправляю все на сервер с помощью push (рис. 4.20).

```

(artzavernyaev@ artzavernyaev)-[~/../study/2023-2024 /Архитектура компьютера/arch-pc]
$ git push
Перечисление объектов: 22, готово.
Подсчет объектов: 100% (22/22), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (16/16), готово.
Запись объектов: 100% (20/20), 310.95 КиБ | 1.97 МиБ/с, готово.
Всего 20 (изменений 1), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To github.com:ArtemZavernyaev/study_2023-2024_arh-pc.git
 abec7af..2f55d30 master -> master

```

Рис. 4.20: Выгрузка изменений на сервер

Проверяю правильность выполнения работы сначала на самом сайте GitHub (рис. 4.21).

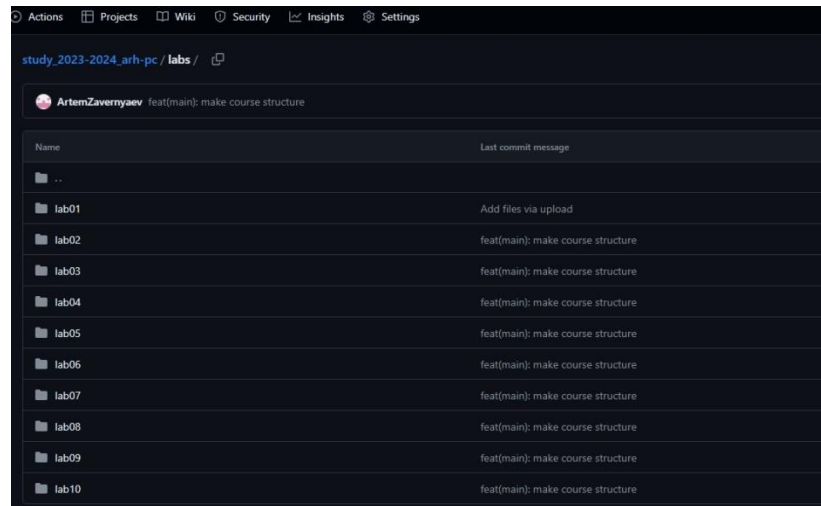


Рис. 4.21: Страница репозитория

4.6 Выполнение заданий для самостоятельной работы

Копирую первую лабораторную с помощью утилиты `cp` и проверяю правильность выполнения команды `cp` с помощью `ls` (рис. 4.22).

```
(artzavernyaev@ artzavernyaev) - [~/../arch-pc/labs/lab01/report]
$ ls
bib image Makefile pandoc report.md Л01_Заверняев_отчёт.pdf
```

Рис. 4.22: Копирование файла

Перехожу в подкаталог `lab01/report` с помощью утилиты `cd`.

Перехожу в директорию, в которой находится отчет по первой лабораторной работе с помощью `cd`. Добавляю файл с отчетом по первой лабораторной работе (рис. 4.23).

```
(artzavernyaev@ artzavernyaev) - [~/../arch-pc/labs/lab01/report]
$ git add Л01_Заверняев_отчёт.pdf
```

Рис. 4.23: Добавление файла на сервер

Отправляю в центральный репозиторий сохраненные изменения командой `git push -f origin master` (рис. 4.24).

```
(artzavernyaev@ artzavernyaev) - [~/../arch-pc/labs/lab01/report]
$ git push -f origin master
Перечисление объектов: 22, готово.
Подсчет объектов: 100% (18/18), готово.
При сжатии изменений используется до 2 потоков
Сжатие объектов: 100% (14/14), готово.
Запись объектов: 100% (14/14), 4.20 МБ | 2.60 МБ/с, готово.
Всего 14 (изменений 6), повторно использовано 0 (изменений 0), повторно использовано пакетов 0
remote: Resolving deltas: 100% (6/6), completed with 2 local objects.
To github.com:ArtemZavernyaev/study_2023-2024_arh-pc.git
+ 53421f5 ... faba3fc master -> master (forced update)
```

Рис. 4.24: Отправка в центральный репозиторий сохраненных изменений

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

Вижу, что отчет по первой лабораторной работе находится в репозитории lab01/report (рис. 4.25).

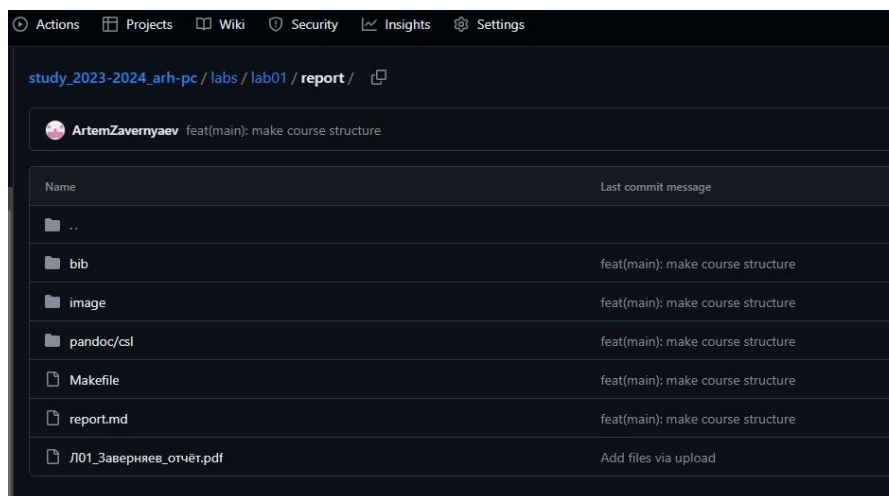


Рис. 4.25: Каталог lab01/report

5 Выводы

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

6 Список литературы

1. Лекционный материал по курсу Архитектура компьютеров