

Отчёт по лабораторной работе №3

Дисциплина: архитектура компьютера

Дворкина Ева Владимировна

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	15
4.6	Настройка каталога курса	17
4.7	Выполнение заданий для самостоятельной работы	20
5	Выводы	27
6	Список литературы	28

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	11
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	14
4.13	Создание рабочего пространства	14
4.14	Страница шаблона для репозитория	15
4.15	Окно создания репозитория	16
4.16	Созданный репозиторий	16
4.17	Перемещение между директориями	17
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	Перемещение между директориями	22
4.33	Копирование файла	23
4.34	Добавление файла на сервер	23
4.35	Перемещение между директориями	23
4.36	Добавление файла на сервер	23
4.37	Подкаталоги и файлы в репозитории	24

4.38 Отправка в центральный репозиторий сохраненных изменений .	24
4.39 Страница каталога в репозитории	24
4.40 Страница последних изменений в репозитории	25
4.41 Каталог lab01/report	25
4.42 Каталог lab02/report	25
4.43 Каталог lab03/report	26

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.

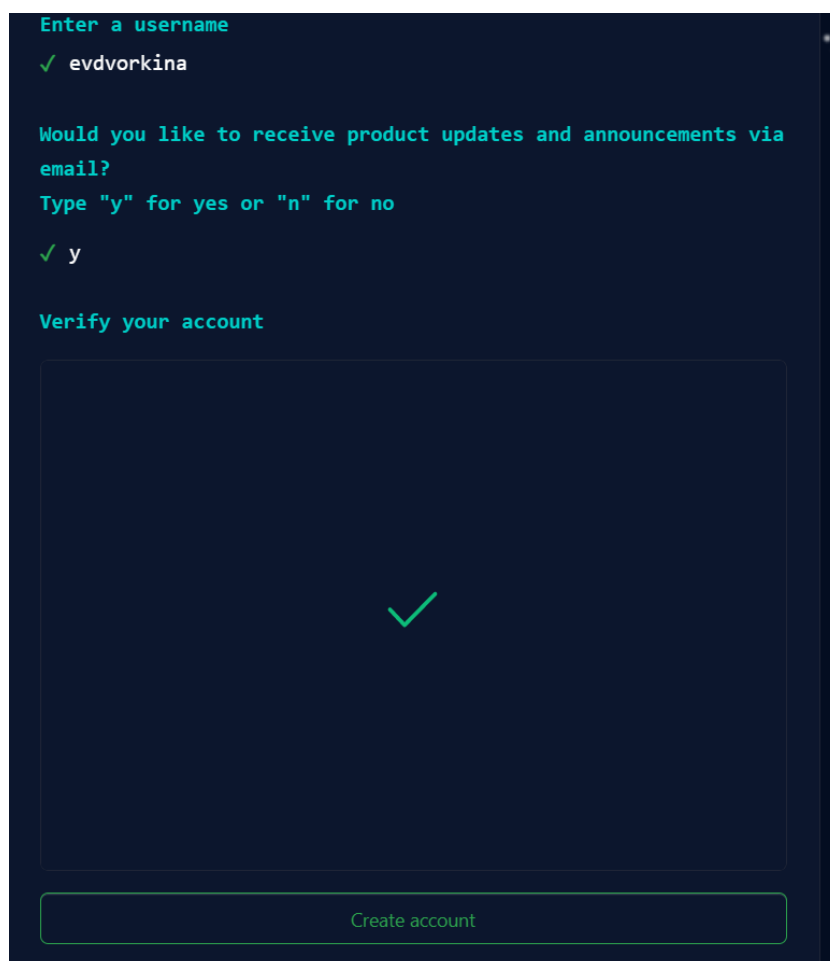
The image shows a dark-themed terminal window with a light blue monospaced font. It displays the steps of creating a GitHub account. First, it prompts 'Enter a username' and shows 'evdvorkina' with a green checkmark. Next, it asks 'Would you like to receive product updates and announcements via email?' and shows 'y' with a green checkmark. Then, it says 'Verify your account' above a large, empty rectangular box. At the bottom, there is a button labeled 'Create account'.

Рис. 4.1: Заполнение данных учетной записи GitHub

Аккаунт создан (рис. 4.2).

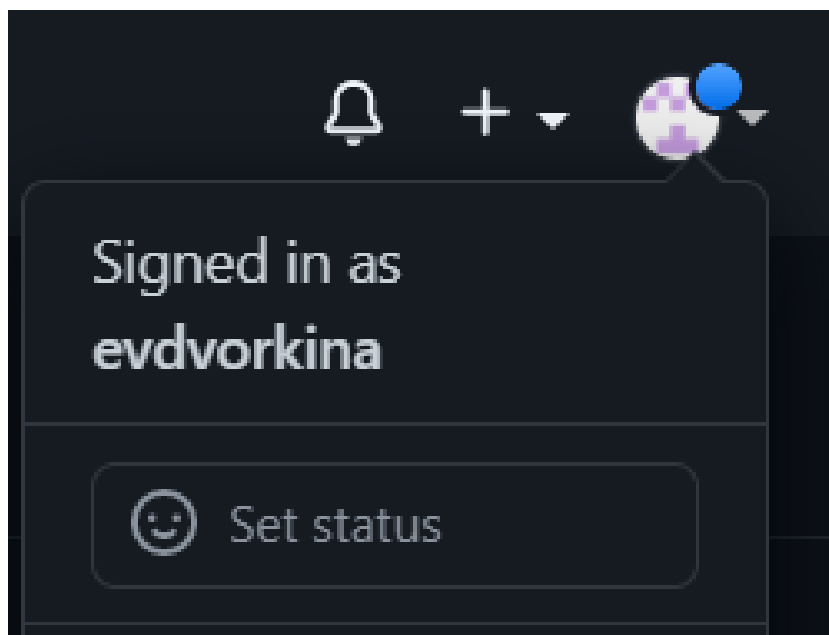


Рис. 4.2: Аккаунт GitHub

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).

```
(evdvorkina@evdvorkina)-[~]  
$ git config --global user.name "<Eva Dvorkina>"  
  
(evdvorkina@evdvorkina)-[~]  
$ git config --global user.email "<1132226447@pfur.ru>"
```

Рис. 4.3: Предварительная конфигурация git

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

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

Рис. 4.4: Настройка кодировки

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

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

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

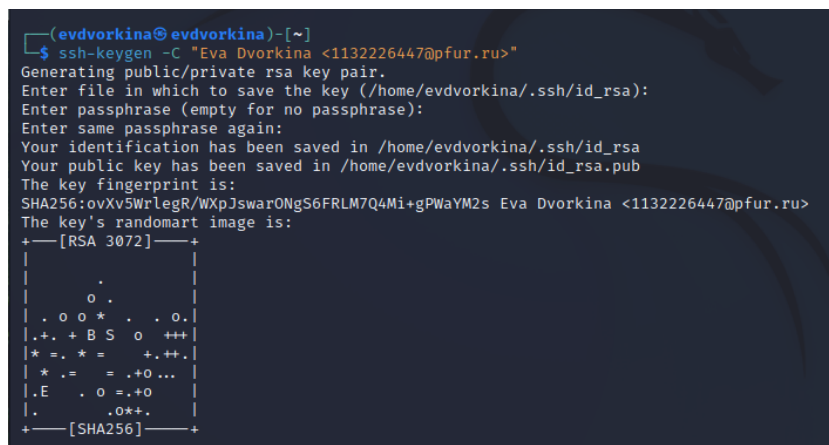
```
(evdvorkina@evdvorkina)-[~]  
$ git config --global core.safecrlf warn
```

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

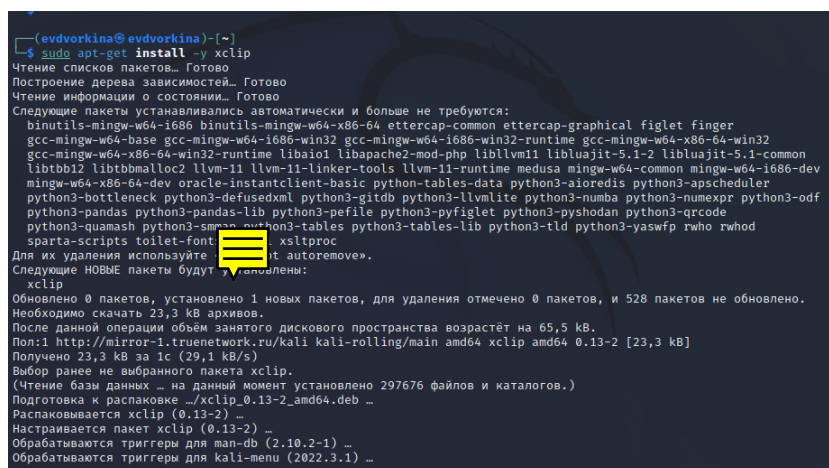
команду `ssh-keygen -C "Имя Фамилия, work@email"`, указывая имя владельца и электронную почту владельца (рис. 4.8). Ключ автоматически сохранится в каталоге `~/.ssh/`.



```
(evdvorkina@evdvorkina)-[~]
$ ssh-keygen -C "Eva Dvorkina <1132226447@pfur.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/evdvorkina/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/evdvorkina/.ssh/id_rsa
Your public key has been saved in /home/evdvorkina/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:ovXv5WrlegR/WXpJswar0NgS6FRLM7Q4Mi+gPWaYM2s Eva Dvorkina <1132226447@pfur.ru>
The key's randomart image is:
+--[RSA 3072]--+
|
|      .
|    o  o  .
|  . o o * . . o .
|+. + B S o +++
|* =. * = .+.+.
| * . = .+o ...
|.E . o =.+o
|. . .o*+.
+--[SHA256]--+
```

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).



```
(evdvorkina@evdvorkina)-[~]
$ sudo apt-get install -y xclip
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Следующие пакеты устанавливались автоматически и больше не требуются:
binutils-mingw-w64-i686 binutils-mingw-w64-x86-64 ettecap-common ettecap-graphical figlet finger
gcc-mingw-w64-base gcc-mingw-w64-i686-win32 gcc-mingw-w64-i686-win32-runtime gcc-mingw-w64-x86-64-win32
gcc-mingw-w64-x86-64-win32-runtime libaio1 libapache2-mod-php libllvm11 liblua5.1-2 liblua5.1-common
libtbb12 libtbbmalloc2 llvm-11 llvm-11-linker-tools llvm-11-runtime medusa mingw-w64-common mingw-w64-i686-dev
mingw-w64-x86-64-dev oracle-instantclient-basic python-tables-data python3-aioredis python3-apscheduler
python3-bottleneck python3-defusedxml python3-gi python3-gi-gdb python3-llvmlite python3-numba python3-numexpr python3-odf
python3-pandas python3-pandas-lib python3-pefile python3-pyfiglet python3-pyshodan python3-qrcode
python3-quasimash python3-smmap python3-tables python3-tables-lib python3-tld python3-yaswfp rwho rwho-d
sparta-scripts toilet-font xclip xclipproc
Для их удаления используйте «sudo apt autoremove».
Следующие НОВЫЕ пакеты будут установлены:
xclip
Обновлено 0 пакетов, установлено 1 новых пакетов, для удаления отмечено 0 пакетов, и 528 пакетов не обновлено.
Необходимо скачать 23,3 kB архивов.
После данной операции объем занятого дискового пространства возрастет на 65,5 kB.
Получено 23,3 kB за 1с (29,1 kB/s)
Выбор ранее не выбранного пакета xclip.
(Чтение базы данных ... на данный момент установлено 297676 файлов и каталогов.)
Подготовка к распаковке ./xclip_0.13-2_amd64.deb ...
Распаковывается xclip (0.13-2) ...
Настраивается пакет xclip (0.13-2) ...
Обрабатываются триггеры для man-db (2.10.2-1) ...
Обрабатываются триггеры для kali-menu (2022.3.1) ...
```

Рис. 4.9: Установка утилиты xclip

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

```
(evdvorkina@evdvorkina)-[~]  
$ cat ~/.ssh/id_rsa.pub | xclip -sel clip
```

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).

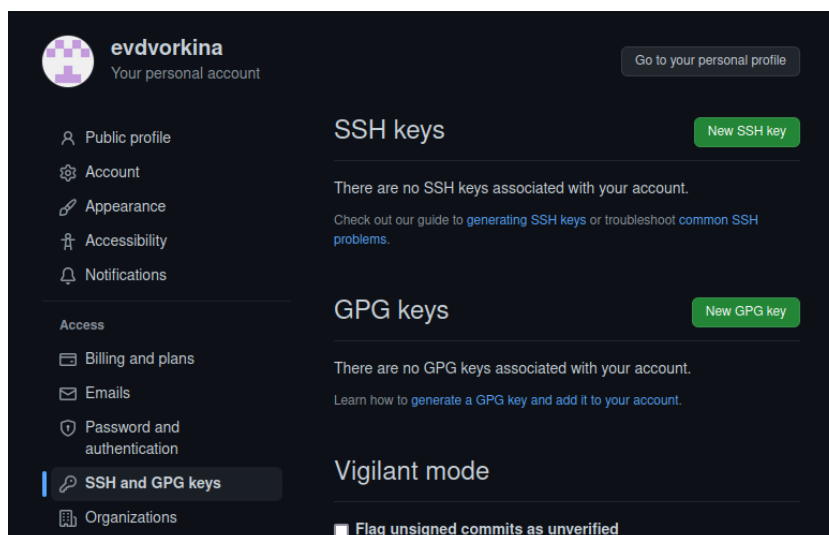


Рис. 4.11: Окно SSH and GPG keys

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

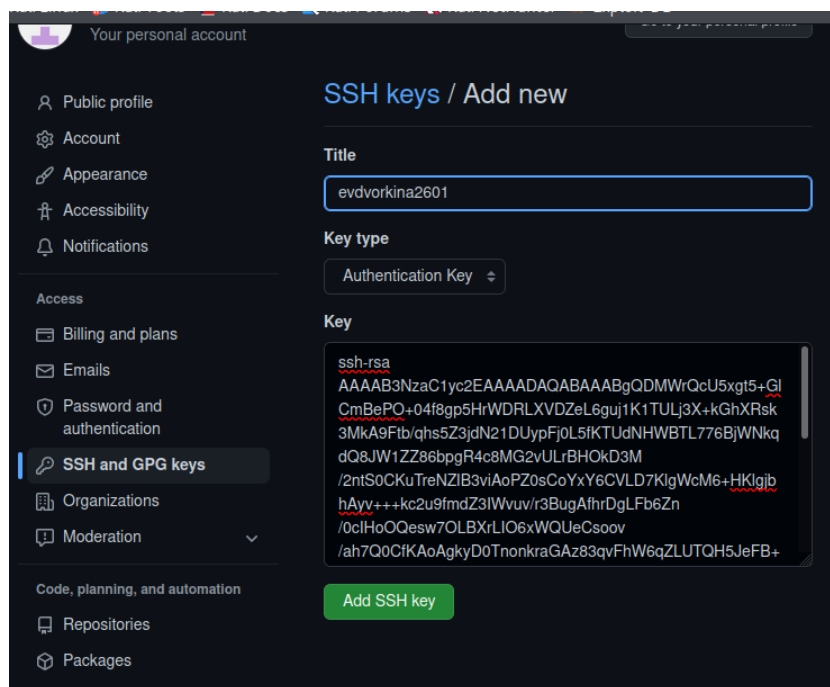


Рис. 4.12: Добавление ключа

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).

```
(evdvorkina@evdvorkina)-[~]
$ mkdir -p work/study/2022-2023/"Архитектура компьютера"

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

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).

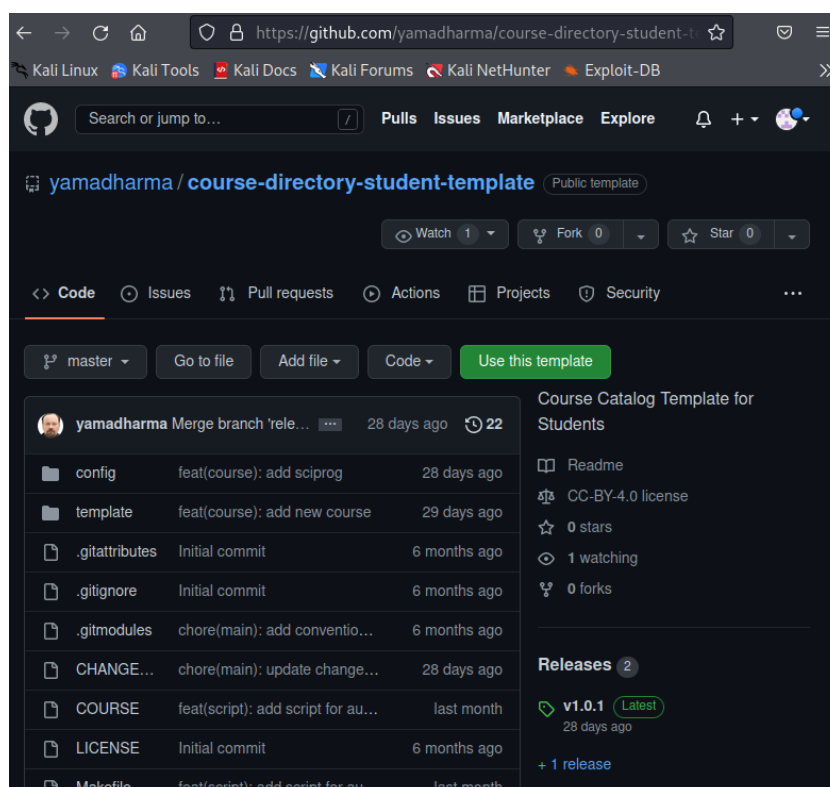


Рис. 4.14: Страница шаблона для репозитория

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

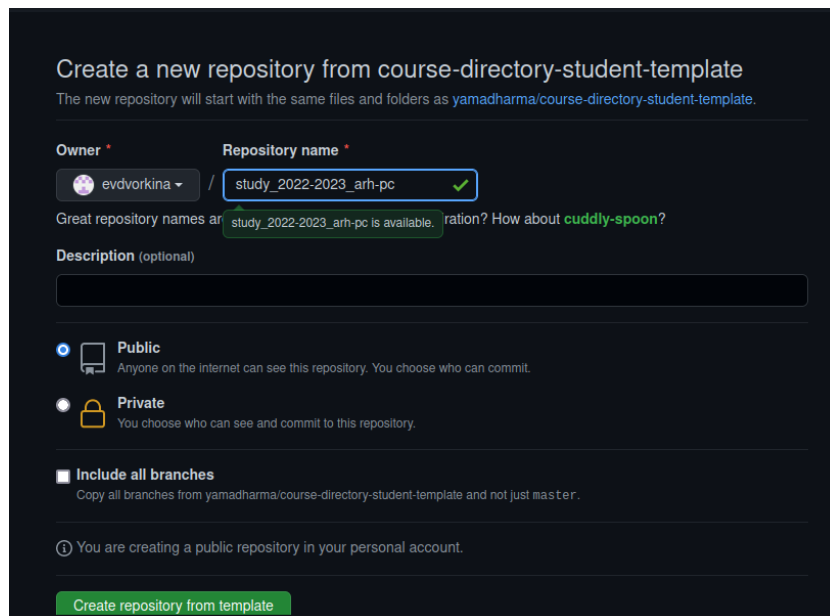


Рис. 4.15: Окно создания репозитория

Репозиторий создан (рис. 4.16).

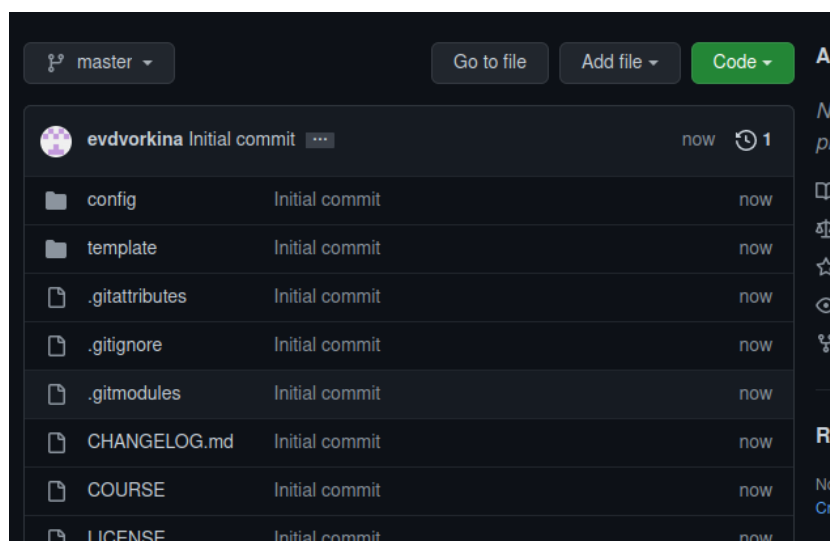


Рис. 4.16: Созданный репозиторий

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


```
(evdvorkina@evdvorkina)-[~]
$ cd ~/work/study/2022-2023/'Архитектура компьютера'
(evdvorkina@evdvorkina)-[~/work/study/2022-2023/Архитектура компьютера]
$
```

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).

```
(evdvorkina@evdvorkina)-[~/work/study/2022-2023/Архитектура компьютера]
$ git clone --recursive git@github.com:evdvorkina/study_2022-2023_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:sh1V3wvV6Tn1JhhpZisF/zLDa0ZBMSvHdkraUlyCnHl.
```

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).

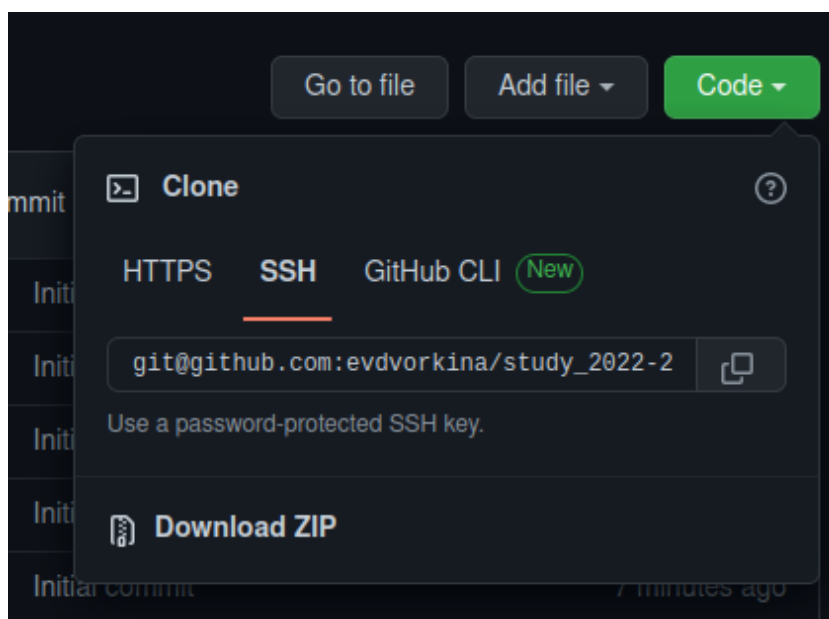


Рис. 4.19: Окно с ссылкой для копирования репозитория

4.6 Настройка каталога курса

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

```
(evdvorkina@evdvorkina)-[~/work/study/2022-2023/Архитектура компьютера]
$ cd ~/work/study/2022-2023/Архитектура\ компьютера/arch-pc

(evdvorkina@evdvorkina)-[~/../study/2022-2023/Архитектура компьютера/arch-pc]
$
```

Рис. 4.20: Перемещение между директориями

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

```
(evdvorkina@evdvorkina)-[~/../study/2022-2023/Архитектура компьютера/arch-pc]
$ rm package.json
```

Рис. 4.21: Удаление файлов

Создаю необходимые каталоги (рис. 4.22).

```
(evdvorkina@evdvorkina)-[~/../study/2022-2023/Архитектура компьютера/arch-pc]
$ echo arch-pc > COURSE

(evdvorkina@evdvorkina)-[~/../study/2022-2023/Архитектура компьютера/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).

```
(evdvorkina@evdvorkina)~[~/../study/2022-2023/Архитектура компьютера/arch-pc]
$ git add .

(evdvorkina@evdvorkina)~[~/../study/2022-2023/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
```

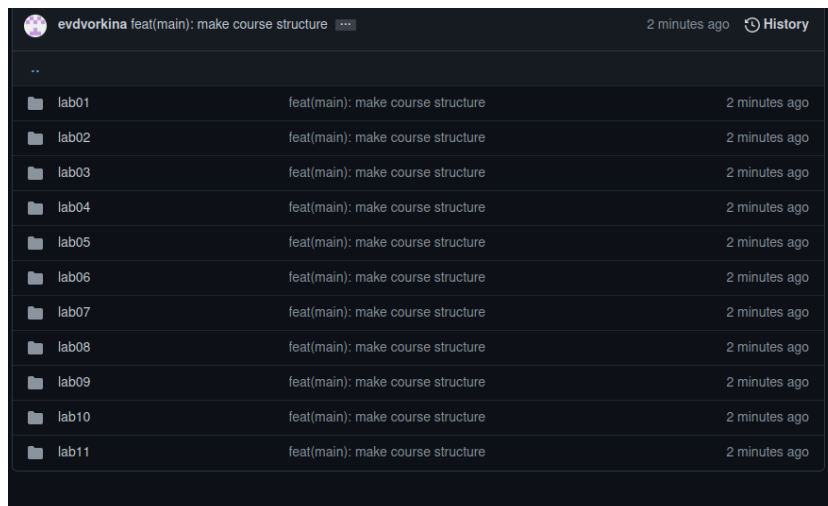
Рис. 4.23: Добавление и сохранение изменений на сервере

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

```
(evdvorkina@evdvorkina)~[~/../study/2022-2023/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 МБ/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:evdvorkina/study_2022-2023_arh-pc.git
 abec7af..2f55d30 master -> master
```

Рис. 4.24: Выгрузка изменений на сервер

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

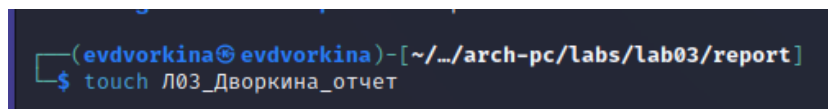


evdvorkina feat(main): make course structure 2 minutes ago History		
..		
lab01	feat(main): make course structure	2 minutes ago
lab02	feat(main): make course structure	2 minutes ago
lab03	feat(main): make course structure	2 minutes ago
lab04	feat(main): make course structure	2 minutes ago
lab05	feat(main): 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

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).



```
(evdvorkina@evdvorkina) - [~/arch-pc/labs/lab03/report]
$ touch Л03_Дворкина_отчет
```

Рис. 4.26: Создание файла

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

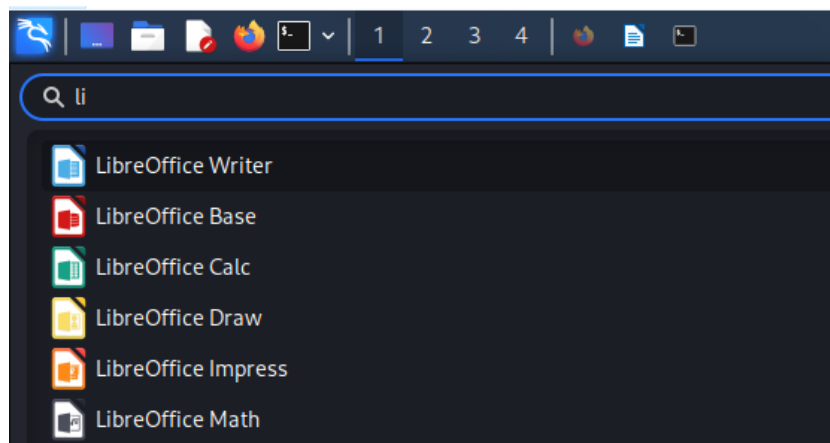


Рис. 4.27: Меню приложений

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

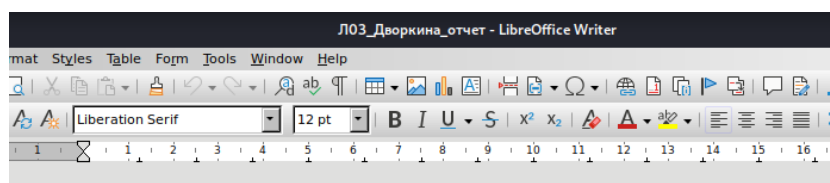


Рис. 4.28: Работа с отчетом в текстовом процессоре

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

```

(evdvorkina@evdvorkina)-[~/arch-pc/labs/lab03/report]
$ cd ..

(evdvorkina@evdvorkina)-[~/Архитектура компьютера/arch-pc/labs/lab03]
$ cd ..

(evdvorkina@evdvorkina)-[~/2022-2023/Архитектура компьютера/arch-pc/labs]
$ cd lab01/

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

(evdvorkina@evdvorkina)-[~/2022-2023/Архитектура компьютера/arch-pc/labs]
$ cd lab01/report/

(evdvorkina@evdvorkina)-[~/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).

```

(evdvorkina@evdvorkina)-[~/arch-pc/labs/lab01/report]
$ ls ~/Загрузки
LibreOffice_7.4.0.3_Linux_x86-64_rpm  Л01_Дворкина_отчет.pdf  Л02_Дворкина_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).

```

(evdvorkina@evdvorkina)-[~/arch-pc/labs/lab01/report]
$ cp ~/Загрузки/Л01_Дворкина_отчет.pdf /home/evdvorkina/work/study/2022-2023/Архитектура компьютера/arch-pc/labs/lab01/report

(evdvorkina@evdvorkina)-[~/arch-pc/labs/lab01/report]
$ ls
bib image Makefile pandoc report report.md Л01_Дворкина_отчет.pdf

```

Рис. 4.31: Копирование файла

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

```

(evdvorkina@evdvorkina)-[~/arch-pc/labs/lab01/report]
$ cd ..

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

(evdvorkina@evdvorkina)-[~/2022-2023/Архитектура компьютера/arch-pc/labs]
$ cd lab02/report

(evdvorkina@evdvorkina)-[~/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).

```
(evdvorkina@evdvorkina) - [~/../arch-pc/labs/lab02/report]
$ cp ~/Загрузки/Л02_Дворкина_отчет.pdf /home/evdvorkina/work/study/2022-2023/'Архитектура компьютера'/arch-pc/labs/lab02/report

(evdvorkina@evdvorkina) - [~/../arch-pc/labs/lab02/report]
$ ls
bib image Makefile pandoc report.md Л02_Дворкина_отчет.pdf
```

Рис. 4.33: Копирование файла

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

```
(evdvorkina@evdvorkina) - [~/../arch-pc/labs/lab02/report]
$ git add Л02_Дворкина_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).

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

(evdvorkina@evdvorkina) - [~/../2022-2023/Архитектура компьютера/arch-pc/labs]
$ cd lab01/report/
```

Рис. 4.35: Перемещение между директориями

Добавляю файл Л01_Дворкина_отчет (рис. 4.36).

```
(evdvorkina@evdvorkina) - [~/../arch-pc/labs/lab01/report]
$ git add Л01_Дворкина_отчет.pdf
```

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).

```

(evdvorkina@evdvorkina)-[~/../study/2022-2023/Архитектура компьютера/arch-pc]
$ cd labs/lab03/report

(evdvorkina@evdvorkina)-[~/../arch-pc/labs/lab03/report]
$ git add Л03_Дворкина_отчет

(evdvorkina@evdvorkina)-[~/../arch-pc/labs/lab03/report]
$ git commit -m "Add existing file"
[master faba3fc] Add existing file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 labs/lab03/report/Л03_Дворкина_отчет

```

Рис. 4.37: Подкаталоги и файлы в репозитории

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

```

(evdvorkina@evdvorkina)-[~/../arch-pc/labs/lab03/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 Миб | 2.60 Миб/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:evdvorkina/study_2022-2023_arh-pc.git
+ 53421f5 ... faba3fc master -> master (forced update)

```

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).

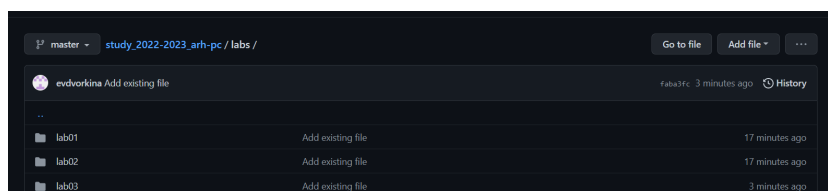


Рис. 4.39: Страница каталога в репозитории

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

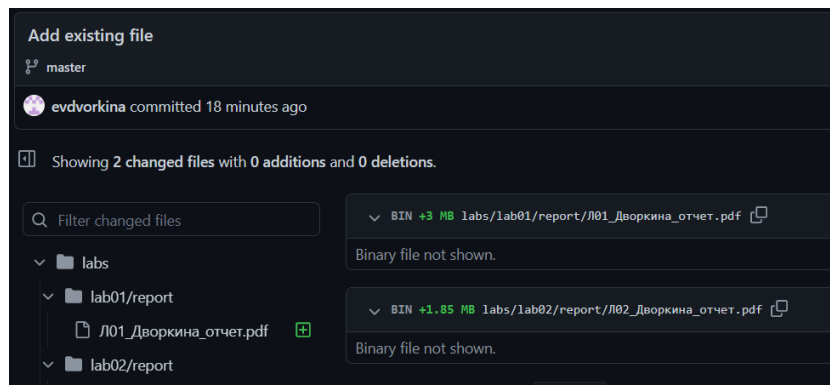


Рис. 4.40: Страница последних изменений в репозитории

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).

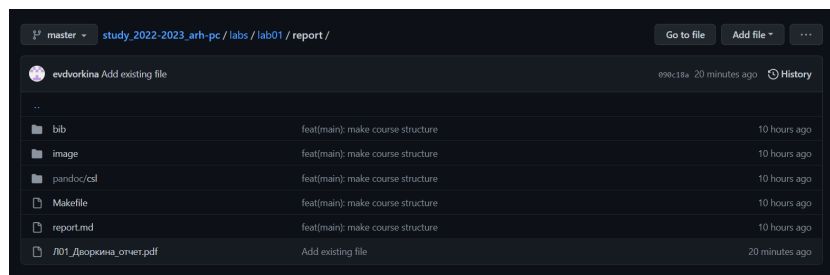


Рис. 4.41: Каталог lab01/report

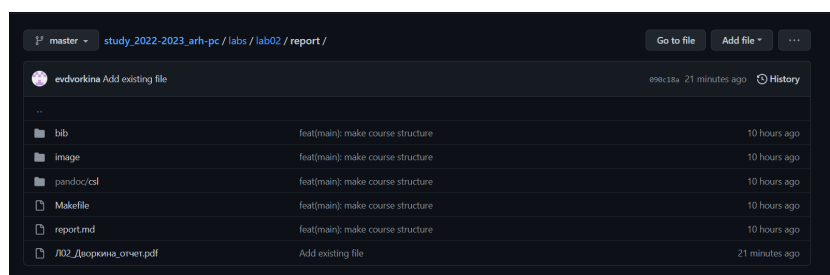


Рис. 4.42: Каталог lab02/report

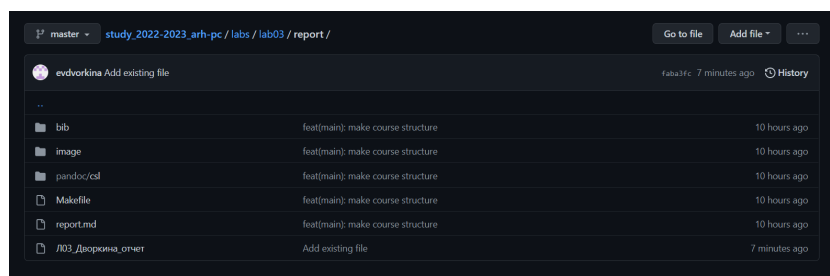


Рис. 4.43: Каталог lab03/report

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