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: Цыкунова Екатерина Михайловна

Студ билет.1132246732

Группа: НКАбд-05-24

МОСКВА

2024 г.

Содержание

Список таблиц

1 Цель работы

Цель данной лабораторной работы - изучение идеологии и применения средств контроля версий, приобретение практических навыков 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

3 Теоретическое введение

Системы контроля версий. Общие понятия

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 про екта. 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 хранили ща 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 кон фликтов, которые могут возникнуть при работе нескольких человек над одним файлом. Можно объединить (слить) изменения, сделанные разными участниками (автоматически или вручную), вручную выбрать нужную версию, отменить изменения вовсе или заблоки ровать 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. Кроме того, 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

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.

Основные команды git

Наиболее часто используемые команды git представлены в таблице 3.1.

Таблица 3.1: Наиболее часто используемые команды Git

Команда	Описание
Git unit	Создание основного дерева репозитория
Git pull	Получение обновлений (изменений) текущего дерева из центрального репозитория
Git push	Отправка всех произведённых изменений локального дерева в центральный репозиторий
Git status	Просмотр списка изменённых файлов в текущей директории
Git diff	Просмотр текущих изменений
Git add	Добавить все изменённые и/или созданные файлы и/или каталоги
Git add имена_файлов	Добавить конкретные изменённые и/или созданные файлы и/или каталоги
Git rm имена_файлов	Удалить файл и/или каталог из индекса репозитория (при этом файл и/или каталог остаётся в локальной директории)
Git commit -am “Описание коммита”	Сохранить все добавленные изменения и все изменённые файлы
Git checkout -b имя_ветки	Создание новой ветки, базирующейся на текущей
Git checkout имя_ветки	Переключение на некоторую ветку (при переключении на ветку, которой ещё нет в локальном репозитории, она будет создана и связана с удалённой)
Git push origin имя_ветки	Отправка изменений конкретной ветки в центральный репозиторий
Git merge --no -ff имя_ветки	Слияние ветки с текущим деревом
Git branch -d имя_ветки	Удаление локальной уже слитой с основным деревом ветки
Git branch -D имя_ветки	Принудительное удаление локальной ветки
Git push origin: имя_ветки	Удаление ветки с центрального репозитория

Стандартные процедуры работы при наличии центрального репозитория

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):

git checkout master

git pull

git checkout -b имя_ветки

Затем можно вносить изменения в локальном дереве и/или ветке. После завершения внесения какого-то изменения в файлы и/или каталоги проекта необ ходимо разместить их в центральном репозитории. Для этого необходимо проверить, какие файлы изменились к текущему моменту:

git status

и при необходимости удаляем лишние файлы, которые не хотим отправлять в центральный репозиторий.

Затем полезно просмотреть текст изменений на предмет соответствия правилам ведения чистых коммитов:

git diff

Если какие-либо файлы не должны попасть в коммит, то помечаем только те файлы, изменения которых нужно сохранить. Для этого используем команды добавления и/или удаления с нужными опциями:

git add имена_файлов

git rm имена_файлов

Если нужно сохранить все изменения в текущем каталоге, то используем:

git add .

Затем сохраняем изменения, поясняя, что было сделано:

git commit -am "Some commit message"

и отправляем в центральный репозиторий:

git push origin имя_ветки или

git push

4 Выполнение лабораторной работы

4.1. Настройка github

Создам учётную запись на сайте https://github.com/ и заполню основные данные.

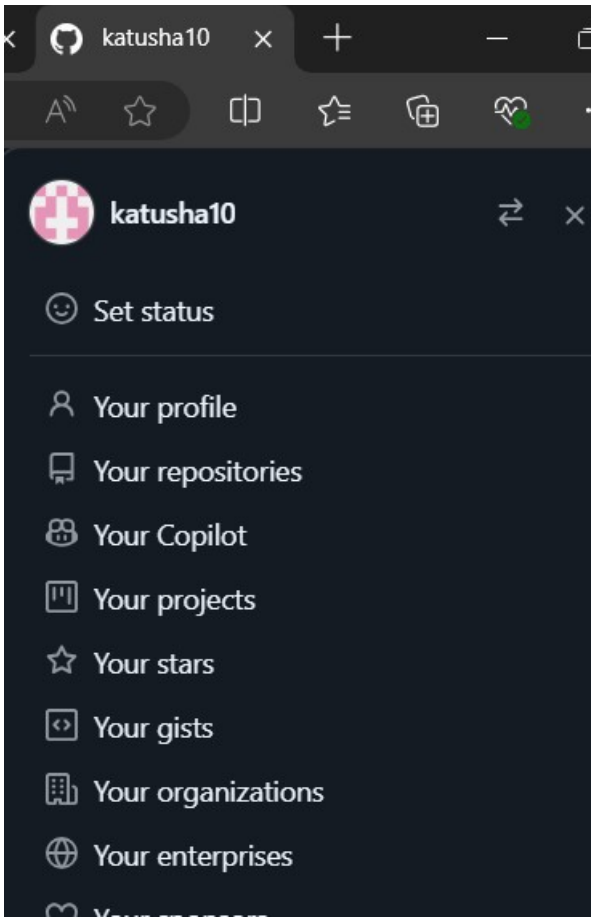


Рис.1. Создание учётной записи

4.2. Базовая настройка git

Сделаю предварительную конфигурацию git. Открою терминал и введу команды, git config --global user.name "", git config --global user.email "" указав свое имя и email, как владельца репозитория.

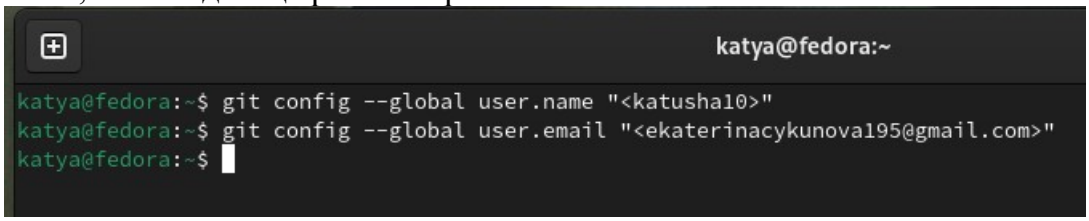


Рис.2. Предварительная кофигурация git

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

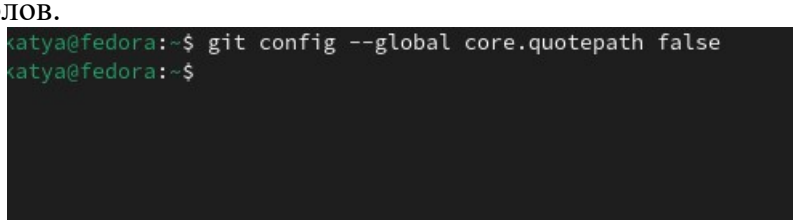


Рис.3. Настройка кодировки

Задам имя “master” для начальной ветки.

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

Рис.4. Создание имени для начальной ветки

Задам параметр autocrlf со значением input (чтобы в системе Linux конвертировать CRLF в LF только при коммитах) и параметр safecrlf со значением warn (Git будет проверять преобразование на обратимость)

```
katya@fedora:~$ git config --global core.autocrlf input
katya@fedora:~$ git config --global core.safecrlf warn
katya@fedora:~$
```

Рис.5. Параметры autocrlf и safecrlf

4.3. Создание SSH ключа

Для последующей идентификации пользователя на сервере репозиториев сгенерирую пару ключей (приватный и открытый). Чтобы сгенерировать их, введу команду ssh-keygen -C “мое имя и фамилия, почта”. Ключ автоматически сохранился в каталоге ~/.ssh/.

```
katya@fedora:~$ ssh-keygen -C "Katya Tsykunova <ekaterinacykunova195@gmail.com>"
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/katya/.ssh/id_ed25519):
Created directory '/home/katya/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/katya/.ssh/id_ed25519
Your public key has been saved in /home/katya/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:fLsz5D+DXznxquUZxJJgiuGBEpa+5KcLPf7B3t//LB0 Katya Tsykunova <ekaterinacykunova195@gmail.com>
The key's randomart image is:
+--[ED25519 256]--+
|  o.             |
| ... .           |
| .. . o   o      |
|  o. . = o . o   |
| o .  o S . o +  |
| .o..    ... oE+ |
|. ooo    o.. .*..|
| o.o o   .=.+=.= |
|  ooo ... o*=**  |
+----[SHA256]-----+
katya@fedora:~$
```

Рис.6. Создание пары ключей

Скопирую из локальной консоли сгенерированный ключ в буфер обмена с помощью команды xclip, которую предварительно установлю.

```
Установить пакет «xclip», предоставляющий команду «xclip»? [N/y] y


 * Ожидание в очереди...
 * Загрузка списка пакетов....
Следующие пакеты должны быть установлены:
 xclip-0.13-21.git11cba61.fc40.x86_64   Command line clipboard grabber
Продолжить с этими изменениями? [N/y] y


 * Ожидание в очереди...
 * Ожидание аутентификации...
 * Ожидание в очереди...
 * Загрузка пакетов...
 * Запрос данных...
 * Проверка изменений...
 * Установка пакетов...
```

Рис.7. Установка команды xclip

Рис.8. Копирование содержимого файла

Далее загружу сгенерённый открытый ключ. Для этого зайду на сайт http: //github.org/ под своей учётной записью и перейду в меню Setting . После этого выберу в боковом меню SSH and GPG keys и нажму кнопку New SSH key .

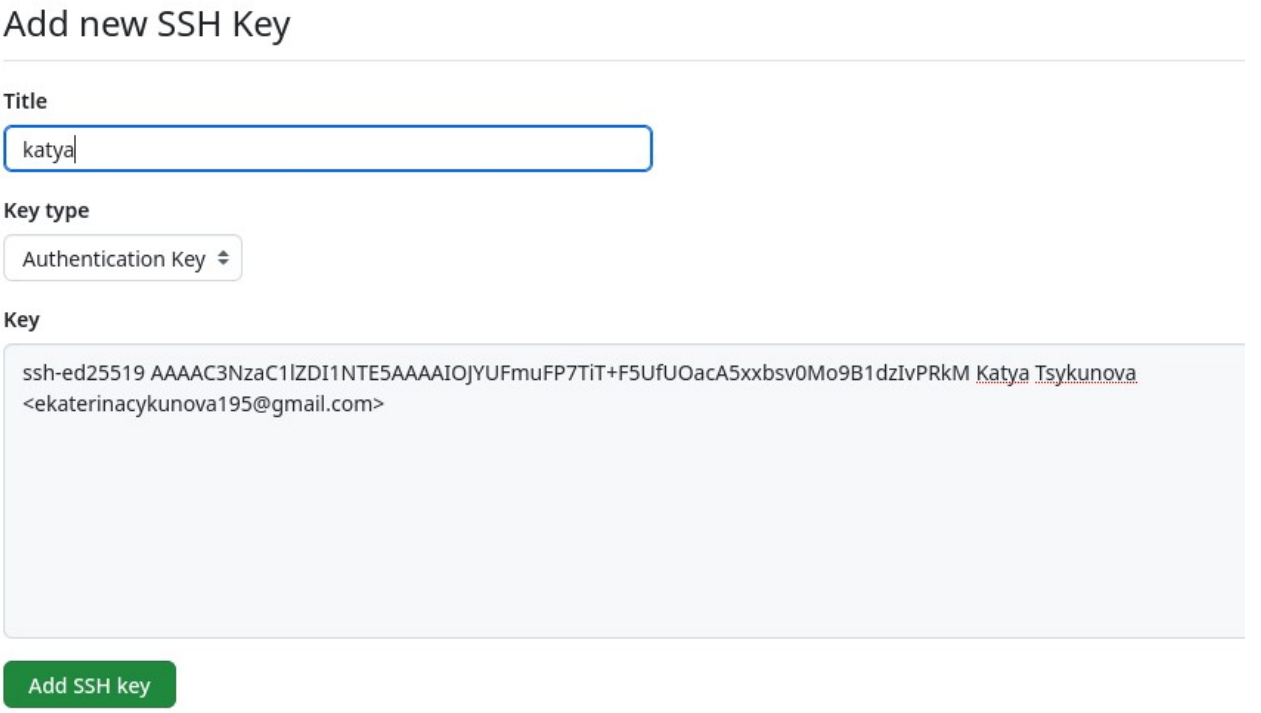


Рис.9. Добавление SSH ключа

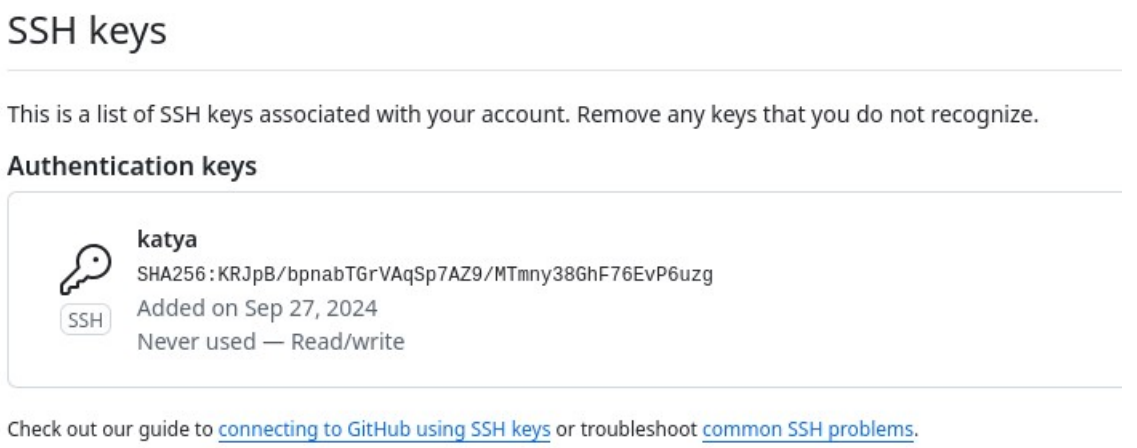


Рис.10. SSH ключ добавлен

4.4. Создание рабочего пространства и репозитория курса на основе шаблона

Теперь расположу рабочее пространство по предмету в определённой иерархии. Закрою браузер и открою терминал. Создам каталог для предмета «Архитектура компьютера» при помощи команды mkdir , используя ключ -p :

mkdir -p ~/work/study/2023-2024/"Архитектура компьютера"

Далее также проверю с помощью команды ls, действительно ли я создала

нужные мне каталоги.

```
katya@fedora:~/.ssh$ mkdir -p ~/work/study/2024-2025/"Архитектура компьютера"
```

Рис.11. Создание каталога

```
katya@fedora:~$ ls
 1   work   Видео   Документы   Загрузки   Изображения   Музыка   Общедоступные   'Рабочий стол'   Шаблоны
katya@fedora:~$
```

Рис. 12. Проверка создания каталога

4.5. Создание репозитория курса на основе шаблона

Перейду на страницу репозитория с шаблоном курса https://github.com/yamadharma/cour se-directory-student-template. Далее выберу Use this template, чтобы использовать этот шаблон для моего репозитория.

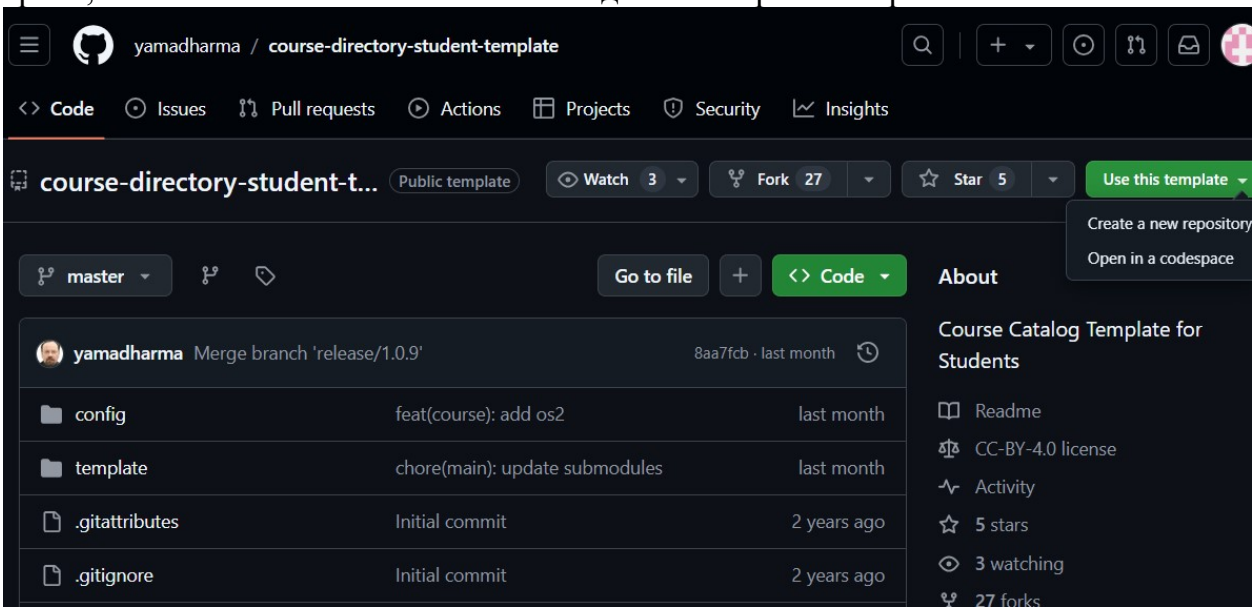


Рис.13. Создание репозитория курса

В открывшемся окне задам имя репозитория (Repository name) study_2024–2025_arh pc и создам репозиторий, нажав на кнопку Create repository from template.

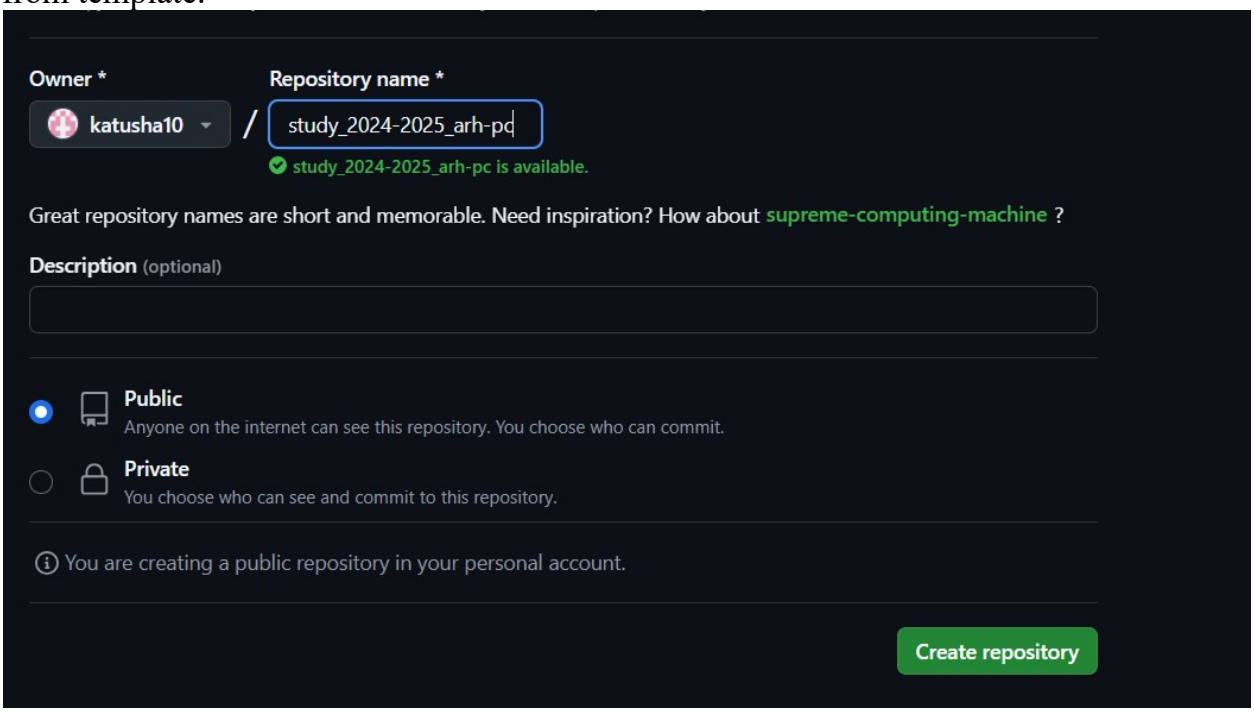


Рис. 14. Репозиторию задаётся имя

Репозиторий успешно создан

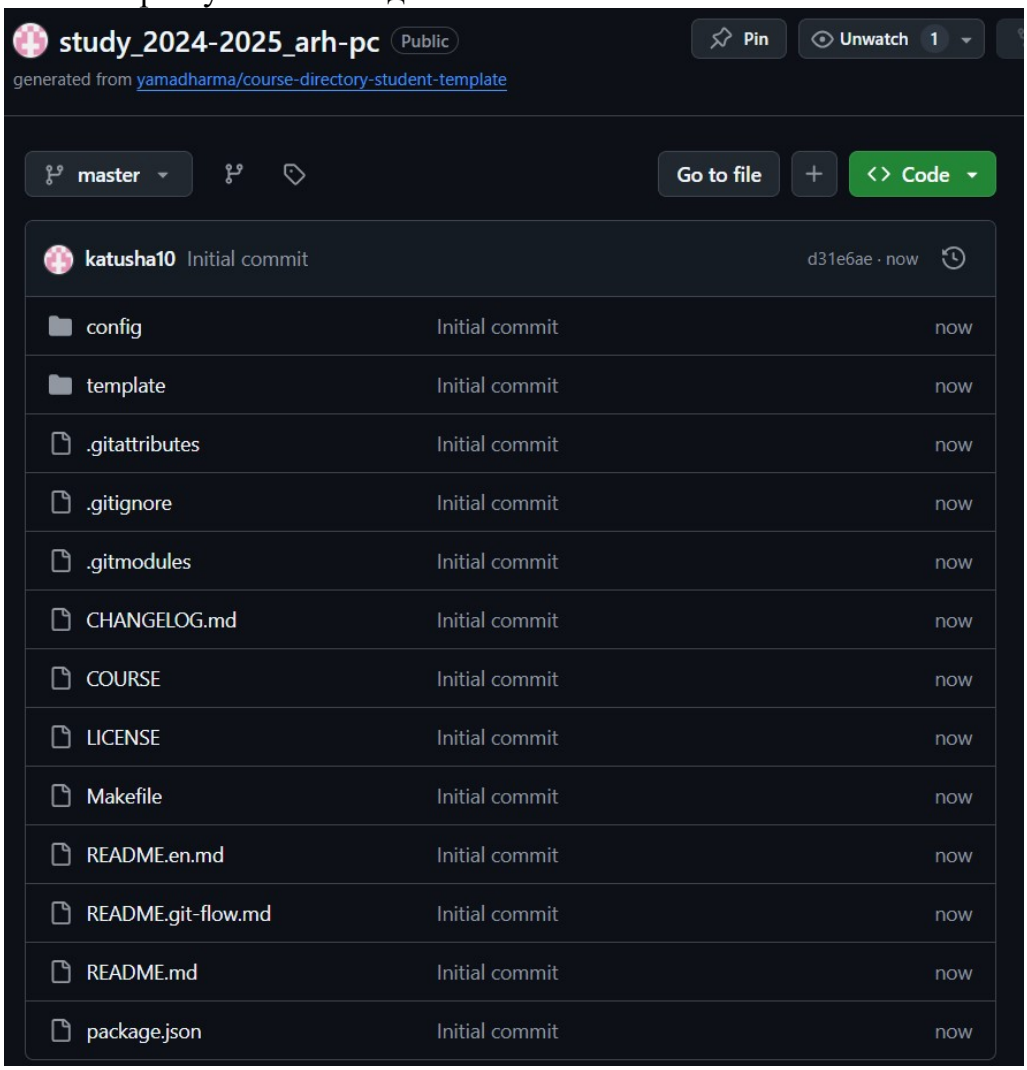


Рис. 15. Проверка создания репозитория

Далее открою терминал и перейду в каталог курса с помощью команды cd

```
katya@fedora:~$ cd ~/work/study/2024-2025/"Архитектура компьютера"
katya@fedora:~/work/study/2024-2025/Архитектура компьютера$
```

Рис.16. Переход в каталог курса

Клонирую созданный репозиторий с помощью команды git clone --recursive git@github.com:/study_2024–2025_arh-pc.git arch-pc

```
katya@fedora:~/work/study/2024-2025/Архитектура компьютера$ git clone --recursive git@github.com:katusha10/study_2024-2025_arh-pc.git arch-pc
Клонирование в «arch-pc»...
remote: Enumerating objects: 33, done.
remote: Counting objects: 100% (33/33), done.
remote: Compressing objects: 100% (32/32), done.
remote: Total 33 (delta 1), reused 18 (delta 0), pack-reused 0 (from 0)
Получение объектов: 100% (33/33), 18.82 КиБ | 385.00 КиБ/с, готово.
Определение изменений: 100% (1/1), готово.
Подмодуль «template/presentation» (https://github.com/yamadharma/academic-presentation-markdown-template.git) зарегистрирован по пути «template/prese
ntation»
Подмодуль «template/report» (https://github.com/yamadharma/academic-laboratory-report-template.git) зарегистрирован по пути «template/report»
Клонирование в «/home/katya/work/study/2024-2025/Архитектура компьютера/arch-pc/template/presentation»...
remote: Enumerating objects: 111, done.
remote: Counting objects: 100% (111/111), done.
remote: Compressing objects: 100% (77/77), done.
remote: Total 111 (delta 42), reused 100 (delta 31), pack-reused 0 (from 0)
Получение объектов: 100% (111/111), 102.17 КиБ | 458.00 КиБ/с, готово.
Определение изменений: 100% (42/42), готово.
Клонирование в «/home/katya/work/study/2024-2025/Архитектура компьютера/arch-pc/template/report»...
remote: Enumerating objects: 142, done.
remote: Counting objects: 100% (142/142), done.
remote: Compressing objects: 100% (97/97), done.
remote: Total 142 (delta 60), reused 121 (delta 39), pack-reused 0 (from 0)
Получение объектов: 100% (142/142), 341.09 КиБ | 1.11 МиБ/с, готово.
Определение изменений: 100% (60/60), готово.
Submodule path 'template/presentation': checked out 'c9b2712b4b2d431ad5086c9c72a02bd2fca1d4a6'
Submodule path 'template/report': checked out 'c26e22effe7b3e0495707d82ef561ab185f5c748'
katya@fedora:~/work/study/2024-2025/Архитектура компьютера$
```

Рис.17. Клонирование созданного репозитория

Затем вставлю ссылку для клонирования со страницы созданного репозитория Code , выбрав вкладку SSH

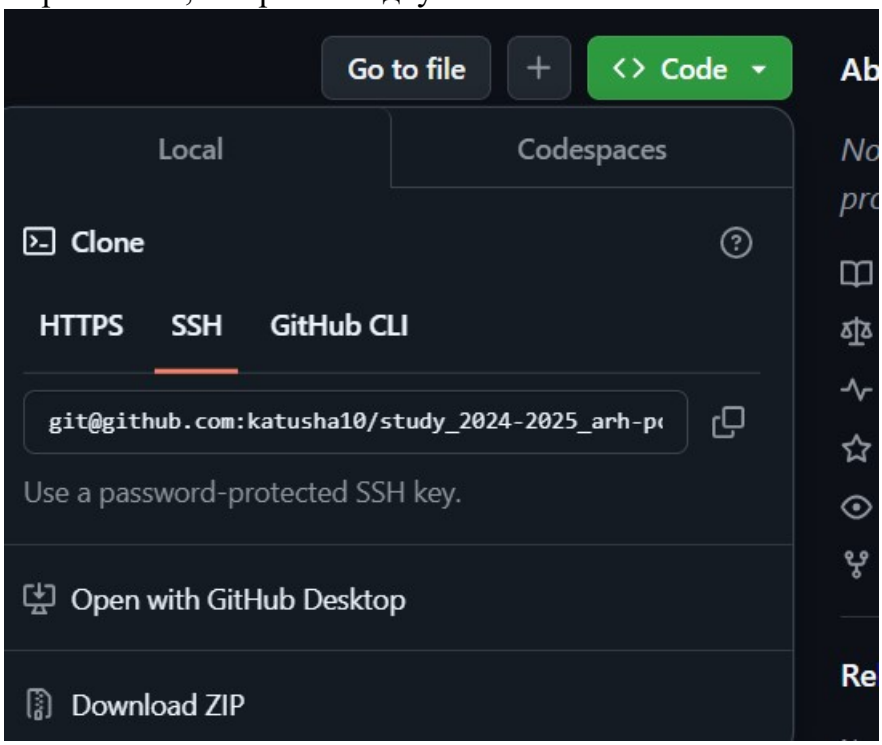


Рис.18. Вставка ссылки для клонирования

4.6. Настройка каталога курса

Перейду в каталог курса с помощью команды cd: cd ~/work/study/2023-2024/"Архитектура компьютера"/arch-pc

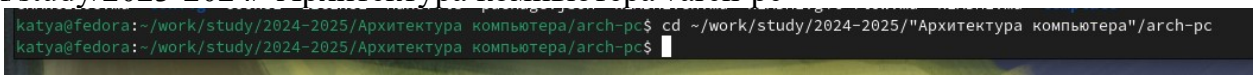


Рис. 19. Переход в каталог курса

Затем удалю лишние файлы с помощью команды rm: rm package.json

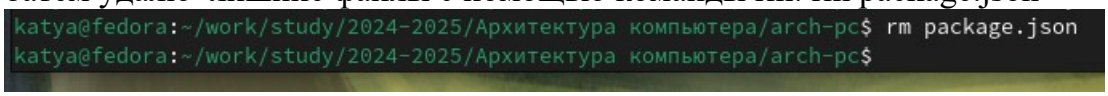


Рис. 20. Удаление лишних файлов

Создам необходимые каталоги с помощью команд echo и make:

 echo arch-pc > COURSE

make

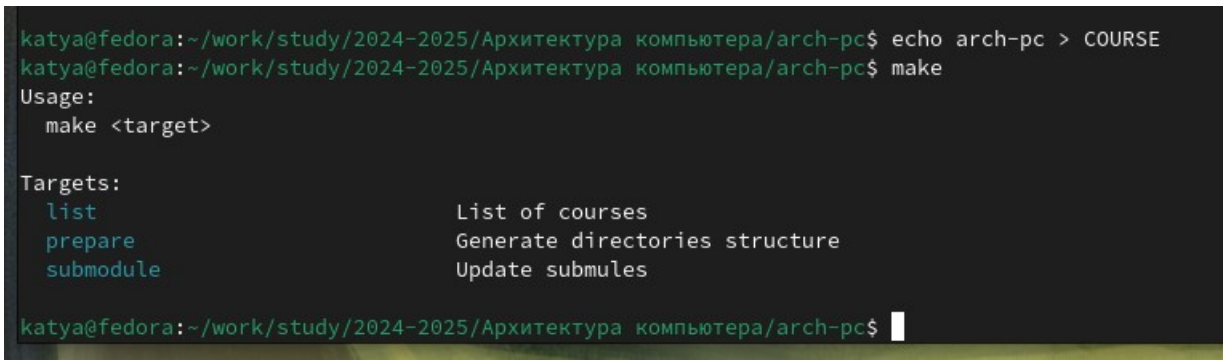


Рис. 21. Создание необходимых каталогов

Отправлю созданные каталоги с локального репозитория на сервер, для этого добавлю все созданные мною каталоги с помощью git add и

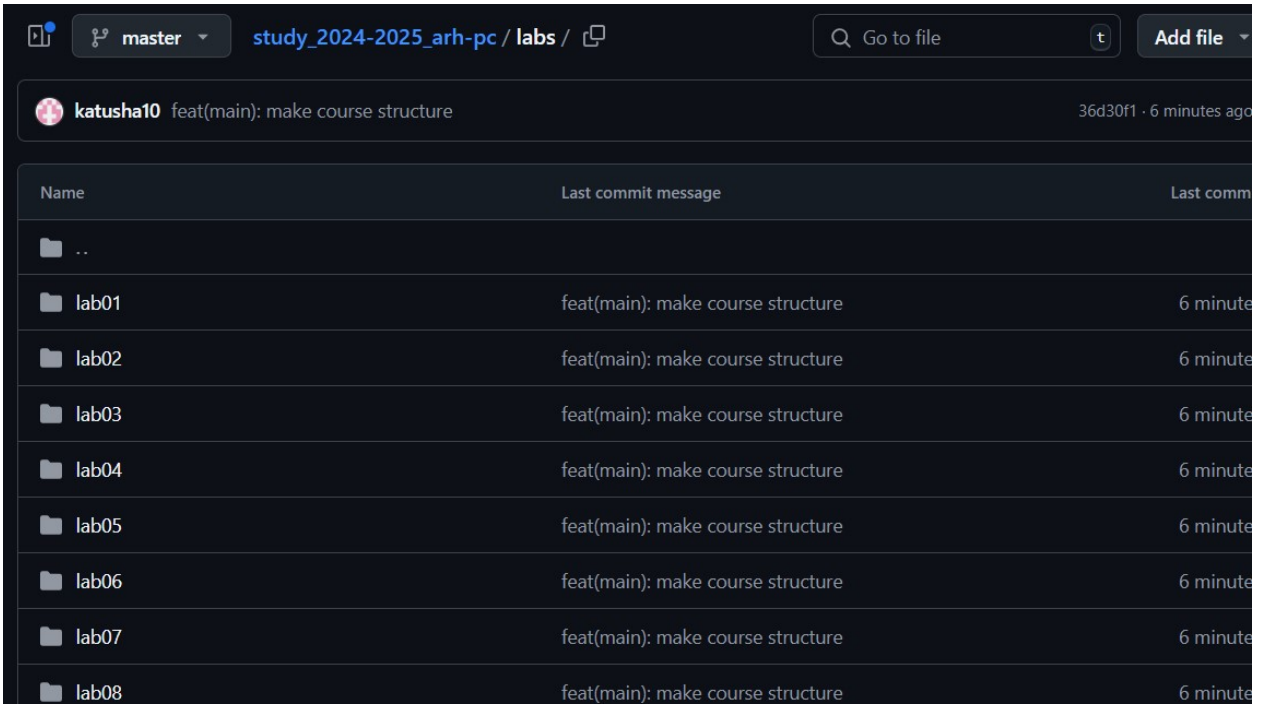
прокомментирую и сохраню изменения на сервере как добавление курса с помощью git commit.

```
katya@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ git add .
katya@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ git commit -am "feat(main): make course structu
[master 36d30f1] feat(main): make course structure
 221 files changed, 53680 insertions(+)
 create mode 100644 labs/README.md
 create mode 100644 labs/README.ru.md
 create mode 100644 labs/lab01/presentation/.projectile
 create mode 100644 labs/lab01/presentation/.texlabroot
 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/.projectile
 create mode 100644 labs/lab02/presentation/.texlabroot
 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
 create mode 100644 labs/lab02/report/report.md
 create mode 100644 labs/lab03/presentation/.projectile
 create mode 100644 labs/lab03/presentation/.texlabroot
 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
```

Отправляю всё на сервер с помощью команды push

```
katya@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc$ git push
Перечисление объектов: 36, готово.
Подсчет объектов: 100% (36/36), готово.
При сжатии изменений используется до 4 потоков
Сжатие объектов: 100% (29/29), готово.
Запись объектов: 100% (35/35), 341.39 КиБ | 2.39 МиБ/с, готово.
Total 35 (delta 4), reused 1 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:katusha10/study_2024-2025_arh-pc.git
   668d0fb..36d30f1  master -> master
katya@fedora:~/work/study/2024-2025/Архитектура компьютера/arch-pc$
```

Проверяю правильность выполнения работы на сайте GitHub


master
study_2024-2025_arh-pc / labs /
Go to file
t
Add file
katusha10 feat(main): make course structure
36d30f1 · 6 minutes ago
Name
Last commit message
Last comm
..
lab01 feat(main): make course structure 6 minute
lab02 feat(main): make course structure 6 minute
lab03 feat(main): make course structure 6 minute
lab04 feat(main): make course structure 6 minute
lab05 feat(main): make course structure 6 minute
lab06 feat(main): make course structure 6 minute
lab07 feat(main): make course structure 6 minute
lab08 feat(main): make course structure 6 minute

5 Выводы

В ходе данной лабораторной работы я изучила идеологию и применение средств контроля версий, приобрела практические навыки по работе с системой git.

Список литературы

1. Архитектура ЭВМ (rudn.ru)