РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 3

дисциплина: Архитектура компьютера

Студент: Замбалова Дина Владимировна

Группа: НПИбд-01-22

МОСКВА

2022 г.

Содержание

Список иллюстраций

Список таблиц

1 Цель работы

Целью работы является изучить идеологию и применение средств контроля версий. Приобрести практические навыки по работе с системой git.

2 Задание

Порядок выполнения работы:

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.

Задание для самостоятельной работы:

1. Создайте отчет по выполнению лабораторной работы в соответствующем каталоге рабочего пространства (labs>lab03>report).

2. Скопируйте отчеты по выполнению предыдущих лабораторных работ в соответствующие каталоги созданного рабочего пространства.

3. Загрузите файлы на github.

3 Теоретическое введение

3.1 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 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 (слить) изменения, сделанные разными участниками (автоматически или вручную), 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. 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.

3.2 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.

3.2 Основные команды git

Наиболее часто используемые команды git представлены в таблице 3.1.

Таблица 3.1: Основные команды git

Команда	Описание
git init	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

3.4. 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 необходимо 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

Порядок выполнения работы:

1. Создаю учётную запись на сайте https://github.com/ и заполняю основные данные (рис. 4.1, 4.2).

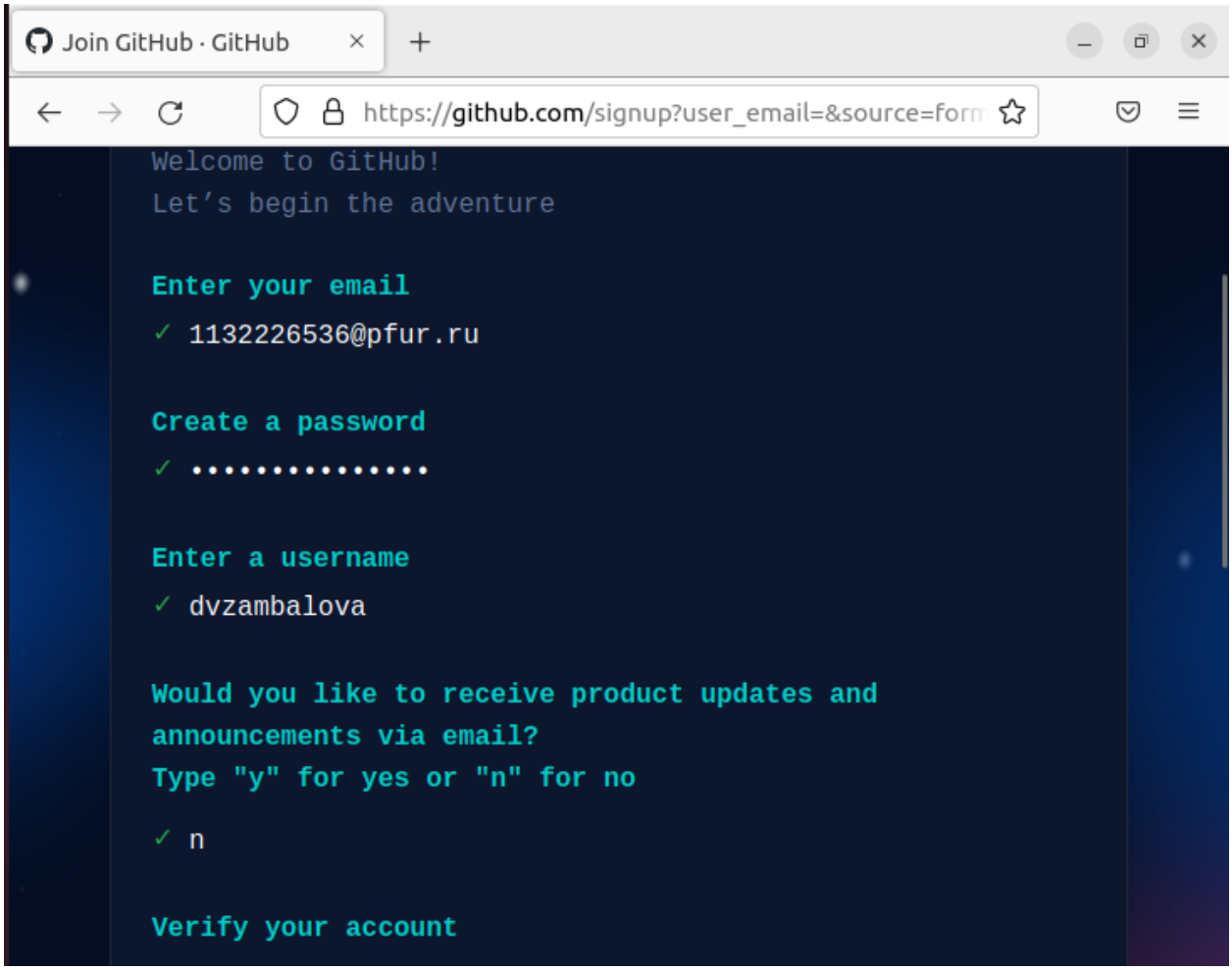

Рис. 4.1: Настройка github

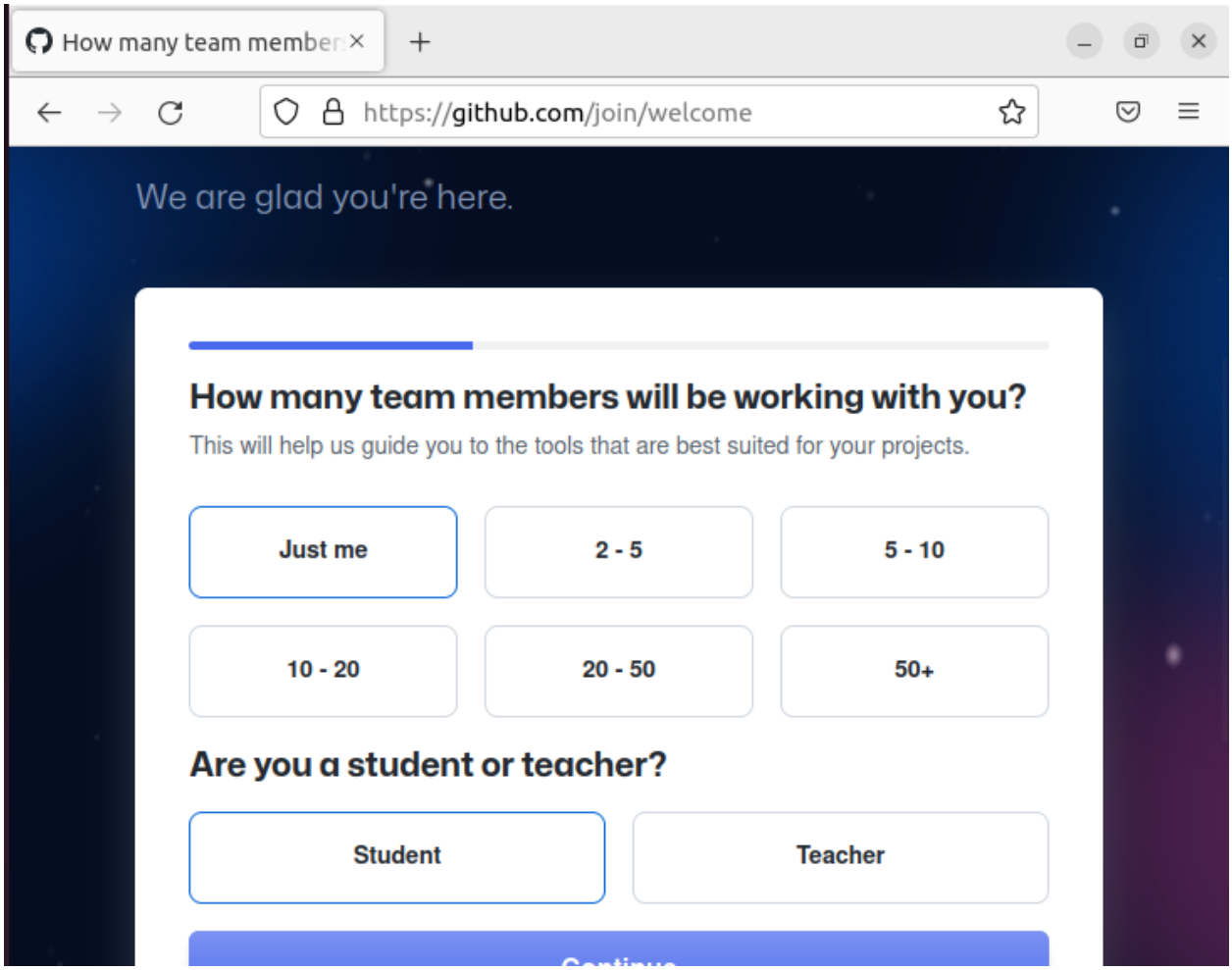

Рис. 4.1: Настройка github

2. Делаю предварительную конфигурацию git. Открываю терминал и ввожу следующие команды, указав имя и email (рис. 4.3).

```
dvzambalova@dvzambalova:~$ git config --global user.name "<dvzambalova>"
dvzambalova@dvzambalova:~$ git config --global user.email "<1132226536@pfur.ru>
"
```

Рис. 4.3: Предварительная конфигурация git

Настраиваю utf-8 в выводе сообщений git (рис. 4.4).

```
dvzambalova@dvzambalova:~$ git config --global core.quotepath false
```

Рис. 4.4: Настройка utf-8 в выводе сообщений git

Задаю имя начальной ветки master (рис. 4.5).

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

Рис. 4.5: Имя начальной ветки

Параметр autocrlf (рис. 4.6).

```
dvzambalova@dvzambalova:~$ git config --global core.autoclif input
```

Рис. 4.6: Параметр autocrlf

Параметр safecrlf (рис. 4.7).

```
dvzambalova@dvzambalova:~$ git config --global core.safeclif warn
```

Рис. 4.7: Параметр safecrlf

3. Для последующей идентификации пользователя на сервере репозиториев генерирую пару ключей (приватный и открытый) (рис. 4.8).

```
dvzambalova@dvzambalova:~$ ssh-keygen -C "dvzambalova <1132226536@pfur.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/dvzambalova/.ssh/id_rsa):
/home/dvzambalova/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/dvzambalova/.ssh/id_rsa
Your public key has been saved in /home/dvzambalova/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:CdzMC3JEWpxYKwgBn5b7PHwLI8e+Od0SL9jrPKBscvg dvzambalova <1132226536@pfur
.ru>
The key's randomart image is:
+---[RSA 3072]----+
|=.   ==.         |
| o +.=o=         |
|  * + * +        |
| . . + o o       |
|  .     S        |
|   =. .          |
| o..X=.+         |
|o =+oB*.o        |
| =E ++==         |
+----[SHA256]-----+
```

Рис. 4.8: Генерирование пары ключей

Ключи сохраняться в каталоге ~/.ssh/. Далее загружаю сгенерированный открытый ключ. Для этого захожу на сайт http://github.org/ под своей учётной записью и перехожу в меню Setting . После этого выбераю в боковом меню SSH and GPG keys и нажимаю кнопку New SSH key . Скопировав из локальной консоли ключ в буфер

обмена cat ~/.ssh/id_rsa.pub | xclip -sel clip, вставляю ключ в появившееся на сайте поле и указываю для ключа имя (рис. 4.9, 4.10).

```
dvzambalova@dvzambalova:~$ cat ~/.ssh/id_rsa.pub | xclip  -sel clip
```

Рис. 4.9: Копирование ключа

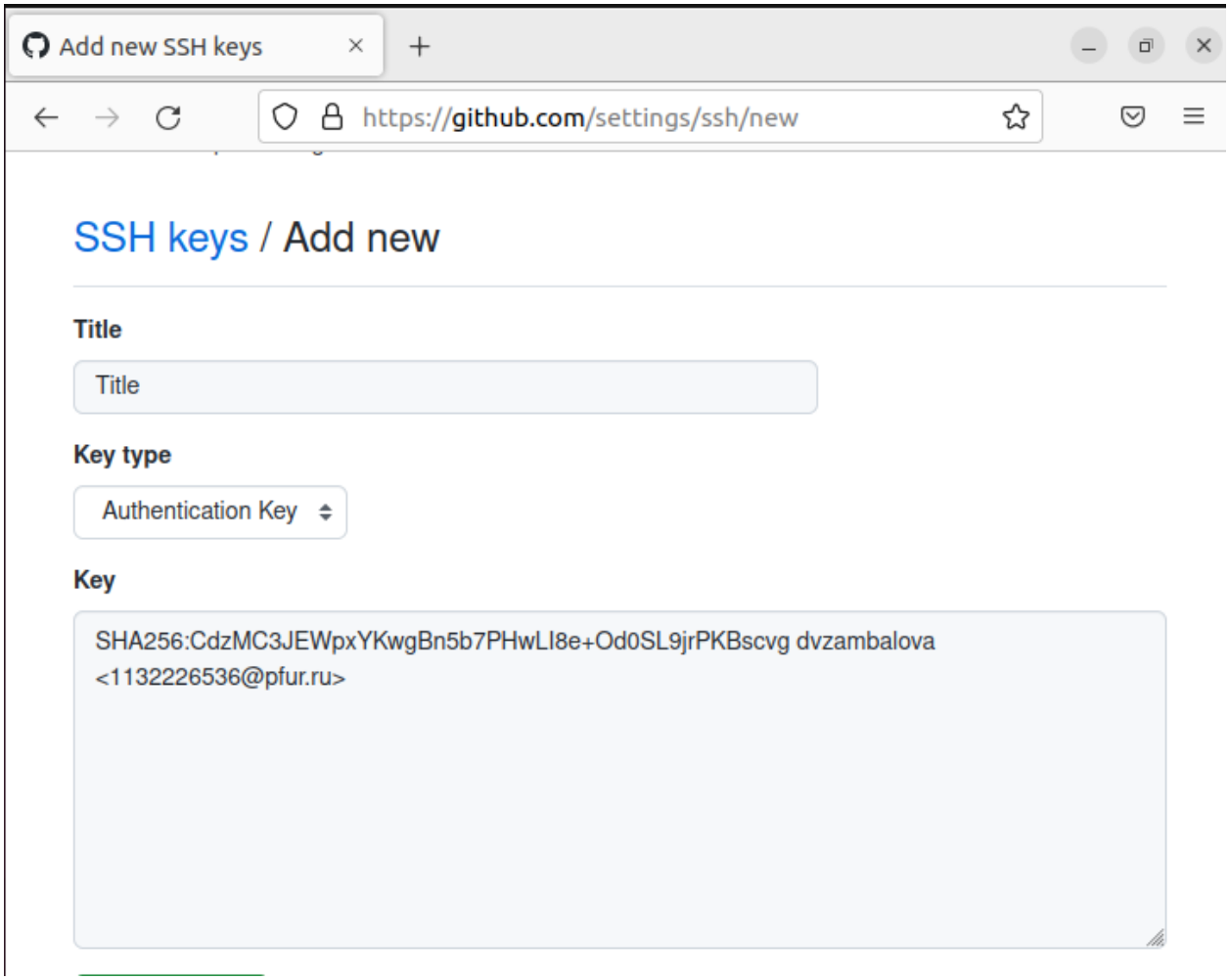

Рис. 4.10: Загрузка сгенерированного открытого ключа

4. Открываю терминал и создаю каталог для предмета «Архитектура компьютера» (рис. 4.11).

```
dvzambalova@dvzambalova:~$ mkdir -p ~/work/study/2022-2023/"Архитектура компьют
ера"
```

Рис. 4.11: Создание каталога для предмета «Архитектура компьютера»

5. Репозиторий на основе шаблона можно создать через web-интерфейс github. Перехожу на станицу репозитория с шаблоном курса https://github.com/yam adharma/course-directory-student-template. Далее выбераю Use this template (рис. 4.12).

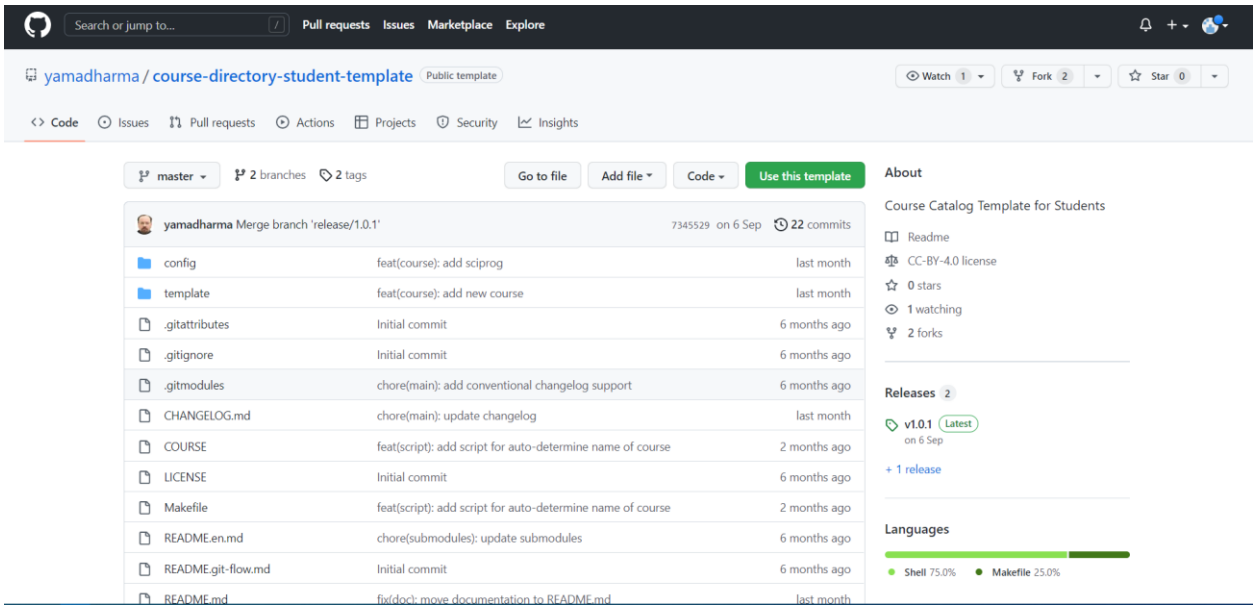

Рис. 4.12: Выбор шаблона

В открывшемся окне задаю имя репозитория study_2022–2023_arh-pc и создаю репозиторий (кнопка Create repository from template) (рис. 4.13).

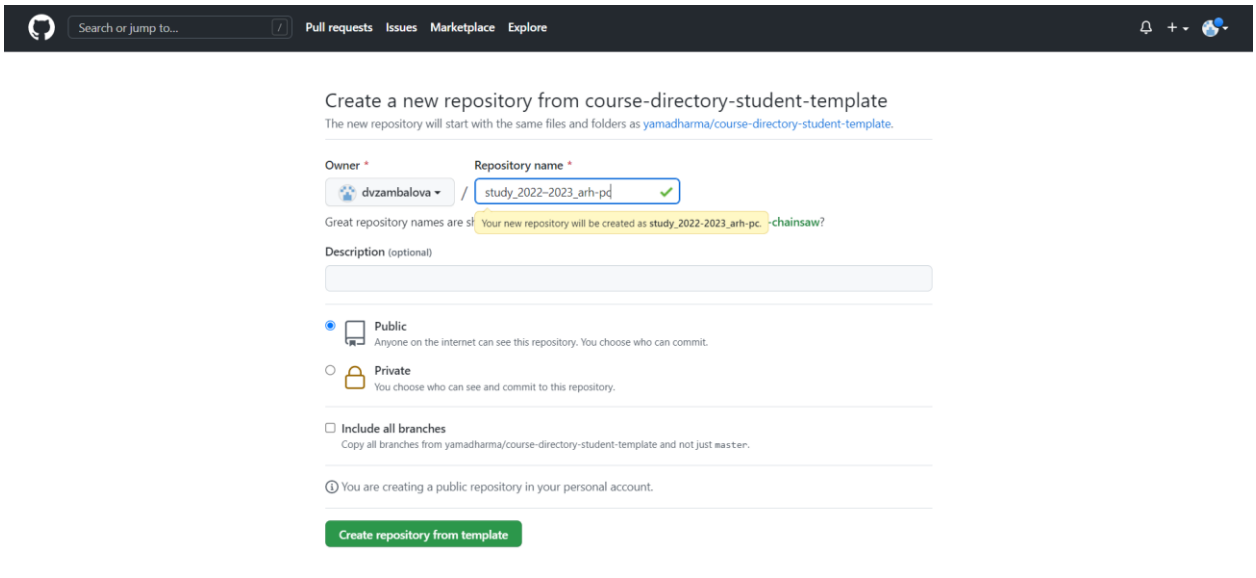

Рис. 4.13: Создание репозитория

Открываю терминал и перехожу в каталог курса (рис. 4.14).

```
dvzambalova@dvzambalova:~$ cd ~/work/study/2022-2023/"Архитектура компьютера"
dvzambalova@dvzambalova:~/work/study/2022-2023/Архитектура компьютера$
```

Рис. 4.14: Переход в каталог курса

Клонирую созданный репозиторий (рис.4.15).

```
dvzambalova@dvzambalova:~/work/study/2022-2023/Архитектура компьютера$ git clon
e --recursive git@github.com:dvzambalova/study_2022-2023_arh-pc.git
Клонирование в «study_2022-2023_arh-pc»…
```

Рис. 4.15: Клонирование репозитория

Ссылку для клонирования можно скопировать на странице созданного репозитория Code -> SSH (рис. 4.16).

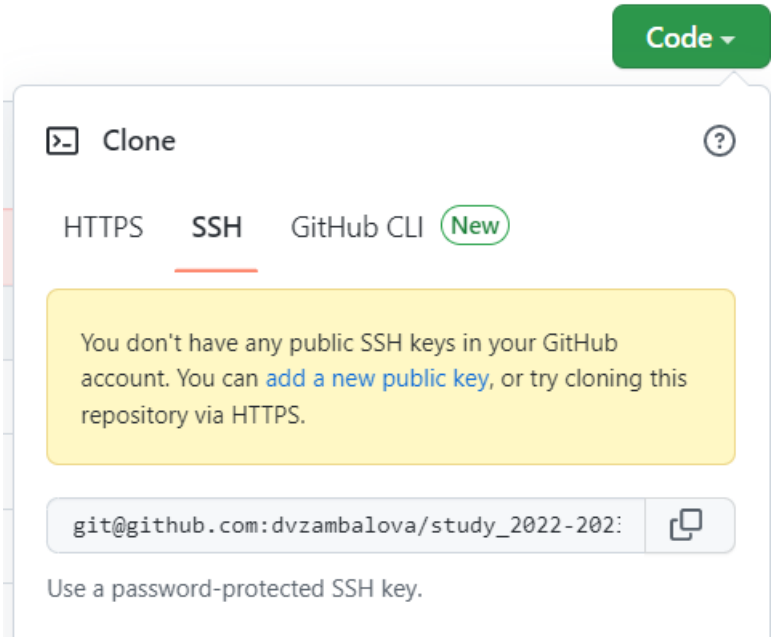

Рис. 4.16: Клонирование репозитория

6. Перехожу в каталог курса и удаляю лишние файлы (рис. 4.17).

```
dvzambalova@dvzambalova:~$ cd ~/work/study/2022-2023/"Архитектура компьютера"/a
rch-pc
dvzambalova@dvzambalova:~/work/study/2022-2023/Архитектура компьютера/arch-pc$
rm package.json
```

Рис. 4.17: Переход в каталог и удаление лишних файлов

Создаю необходимые каталоги (рис. 4.18).

Рис. 4.18: Создание необходимых каталогов

Отправляю файлы на сервер (рис 4.19, 4.20, 4.21).

Рис. 4.19: Отправка файлов на сервер

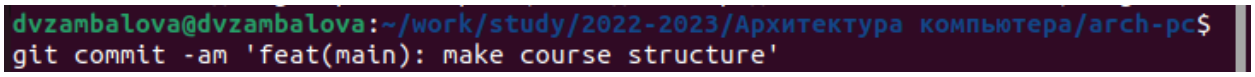

Рис. 4.20: Отправка файлов на сервер

Рис. 4.21: Отправка файлов на сервер

Проверяю правильность создания иерархии рабочего пространства в локальном репозитории и на странице github.

Задание для самостоятельной работы:

1.

5 Выводы

Список литературы

1. Колисниченко Д.Н. Самоучитель. Linux 2-е изд. СПб.: БХВ-Петербург, 2008.
2. Кофлер М. Весь Linux. Установка, конфигурирование, использование. 7-е изд. М.: Бином: Бином-Пресс, 2007.
3. Граннеман С. Linux: карманный справочник: необходимый код и команды. М. и др.: Вильямс, 2007.
4. Основные Linux команды для новичка. URL: https://habr.com/ru/post/501442/. (Дата обращения: 27.09.22)