

Отчёт по лабораторной работе No2

Дисциплина: архитектура компьютера

Хасанов Тимур

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	25

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	Добавление файла на сервер	22
4.33	Подкаталоги и файлы в репозитории	23
4.34	Отправка в центральный репозиторий сохраненных изменений	23
4.35	Страница каталога в репозитории	23
4.36	Страница последних изменений в репозитории	23
4.37	Каталог lab01/report	24

4.38 Каталог lab02/report	24
4.39 Страница каталога в репозитории	24

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

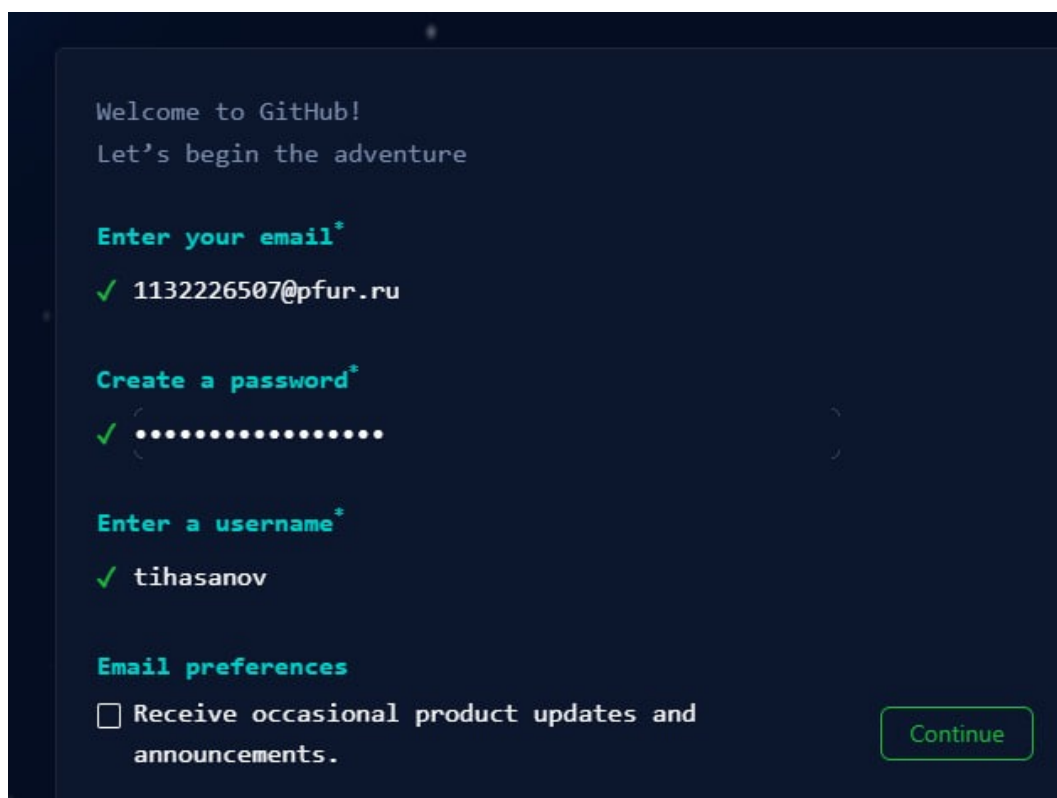
Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля 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 локальное дерево 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 заполнил основные данные учетной записи.



The image shows a dark-themed GitHub sign-up interface. At the top, it says "Welcome to GitHub!" and "Let's begin the adventure". Below this are three input fields, each with a green checkmark indicating successful completion:

- Enter your email***: The email address "1132226507@pfur.ru" is entered.
- Create a password***: The password field is filled with dots.
- Enter a username***: The username "tihasanov" is entered.

Below the username field is the **Email preferences** section, which includes a checkbox labeled "Receive occasional product updates and announcements." that is currently unchecked. A green "Continue" button is located at the bottom right of the form.

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).

```
[tihasanov@tihasanov ~]$ git config --global user.name "<Timur Hasanov>"  
[tihasanov@tihasanov ~]$ git config --global user.email "<1132226507@pfur.ru>"
```

Рис. 4.3: Предварительная конфигурация git

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

```
[tihasanov@tihasanov ~]$ git config --global core.quotePath false
```

Рис. 4.4: Настройка кодировки

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

```
[tihasanov@tihasanov ~]$ 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.

```
[tihasanov@tihasanov ~]$ 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.

```
[tihasanov@tihasanov ~]$ 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/`.

```
[tihasanov@tihasanov ~]$ ssh-keygen -C "Timur Hasanov <1132226507@pfur.ru>"
Generating public/private rsa key pair.
Enter file in which to save the key (/home/tihasanov/.ssh/id_rsa):
Created directory '/home/tihasanov/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/tihasanov/.ssh/id_rsa.
Your public key has been saved in /home/tihasanov/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:rKEb8uuu/SK9A3L066mnBGk8t657EGJbn8Lvck0GRqs Timur Hasanov <1132226507@pfur.ru>
The key's randomart image is:
+---[RSA 2048]-----+
|
|o+ . .
|+=.. o S
|++=0.+ 0
|.*==0.
|.*BB+.
|EBX0XB.
+-----[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 `-u` от имени суперпользователя, введя в начале команды `sudo` (рис. 4.9).

```
[tihasanov@tihasanov ~]$ sudo yum install -y xclip
Загружены модули: fastestmirror, langpacks
Заблокировано /var/run/yum.pid: другая копия запущена с pid 19655.
Another app is currently holding the yum lock; waiting for it to exit...
Другое приложение: PackageKit
Память : 35 M RSS (928 MB VSZ)
Запущено : Mon Jun 3 00:16:46 2024 – 00:12 назад
Статус : Запуск, pid: 19655
Another app is currently holding the yum lock; waiting for it to exit...
Другое приложение: PackageKit
Память : 52 M RSS (944 MB VSZ)
Запущено : Mon Jun 3 00:16:46 2024 – 00:14 назад
Статус : Запуск, pid: 19655
Another app is currently holding the yum lock; waiting for it to exit...
Другое приложение: PackageKit
Память : 62 M RSS (954 MB VSZ)
Запущено : Mon Jun 3 00:16:46 2024 – 00:16 назад
Статус : Запуск, pid: 19655
Another app is currently holding the yum lock; waiting for it to exit...
Другое приложение: PackageKit
Память : 82 M RSS (973 MB VSZ)
Запущено : Mon Jun 3 00:16:46 2024 – 00:18 назад
Статус : Запуск, pid: 19655
Another app is currently holding the yum lock; waiting for it to exit...
```

Рис. 4.9: Установка утилиты xclip

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

```
[tihasanov@tihasanov ~]$ 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).

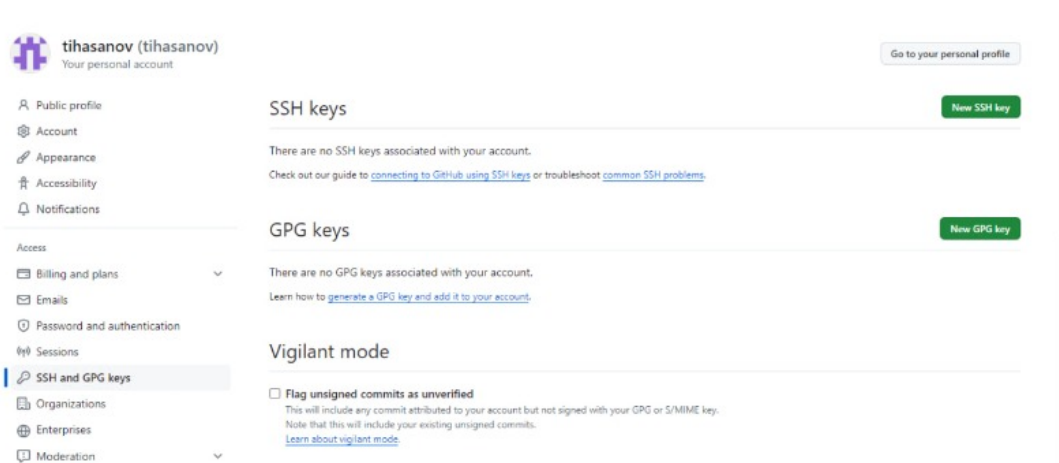


Рис. 4.11: Окно SSH and GPG keys

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

tihasanov2007

Key type

Authentication Key

Key

```
ssh-rsa
AAAAB3NzaC1yc2EAAAABIwAAAQEAKlOUpkDHrfHY17SbrmTlpNLTGK9Tjom/BWDSU
GPI+nafzIHDTYW7hdl4yZ5ew18JH4JW9jbhUFrviQzM7xlELEVf4h9IFX5QVkbPppSwg0cda3
Pbv7kOdJ/MTyBIWXFCR+HAo3FXRitBqxiX1nKhXpHAZsMciLq8V6RjsNAQwdsdMFvSIVK/7X
A
t3FaoJoAsncM1Q9x5+3V0Ww68/eIFmb1zuUFliQJKorrX88XvpNDviYNbv6vw/Pb0rwert/En
```

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).

```
[tihasanov@tihasanov ~]$ mkdir -p work/study/2023-2024/"Архитектура компью
реpa"
[tihasanov@tihasanov ~]$ ls
work  Документы  Изображения  Общедоступные  Шаблоны
видео  Загрузки  Музыка      Рабочий стол
[tihasanov@tihasanov ~]$
```

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/yamadharm/course-directory-student-template>. Далее выбираю «Use this template», чтобы использовать этот шаблон для своего репозитория (рис. 4.14).

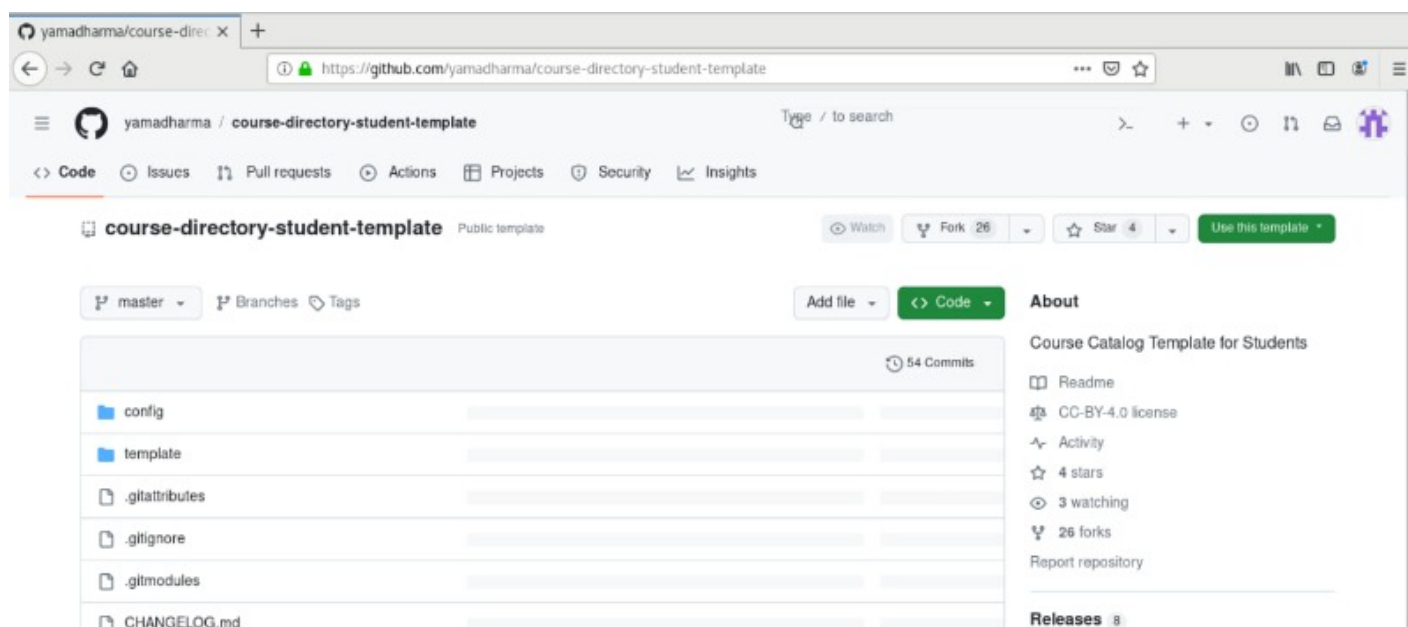


Рис. 4.14: Страница шаблона для репозитория

В открывшемся окне задаю имя репозитория (Repository name): `study_2023–2024_arh-` и создаю репозиторий, нажимаю на кнопку «Create repository from template» (рис. 4.15).

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Required fields are marked with an asterisk (*).


Owner * tihasanov / Repository name * study_2023-2024_arh-pc


✓ Your new repository will be created as -study_2023-2024_arh-pc.
The repository name can only contain ASCII letters, digits, and the characters -, ., and _.

Great repository names are short and memorable. Need inspiration? How about [psychic-happiness](#) ?

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. 4.15: Окно создания репозитория

Репозиторий создан (рис. 4.16).

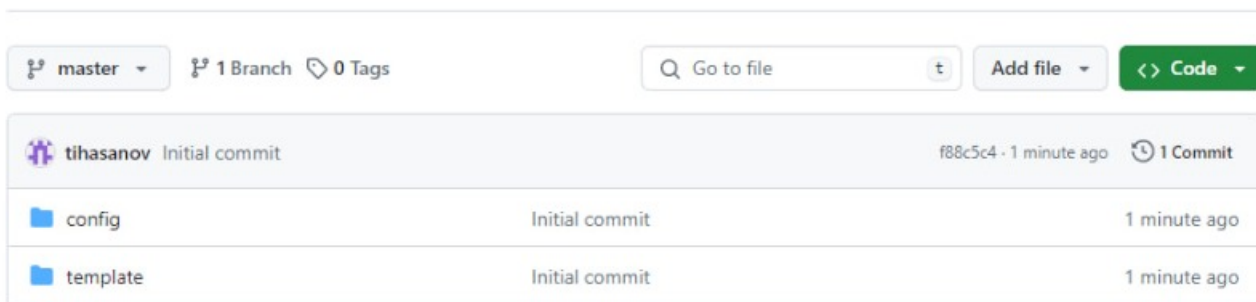


Рис. 4.16: Созданный репозиторий

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


```
[tihasanov@tihasanov ~]$ cd ~/work/study/2023-2024/"Архитектура компьютера"
[tihasanov@tihasanov Архитектура компьютера]$
```

Рис. 4.17: Перемещение между директориями

Клонирую созданный репозиторий с помощью команды `git clone --recursive git@github.com:/study_2023-2024_arh-pc.git arch-pc` (рис. 4.18).

```
[tihasanov@tihasanov Архитектура компьютера]$ git clone --recursive git@github.com:tihasanov/-study_2023-2024_arh-pc.git arch-pc
Cloning into 'arch-pc'...
```

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).

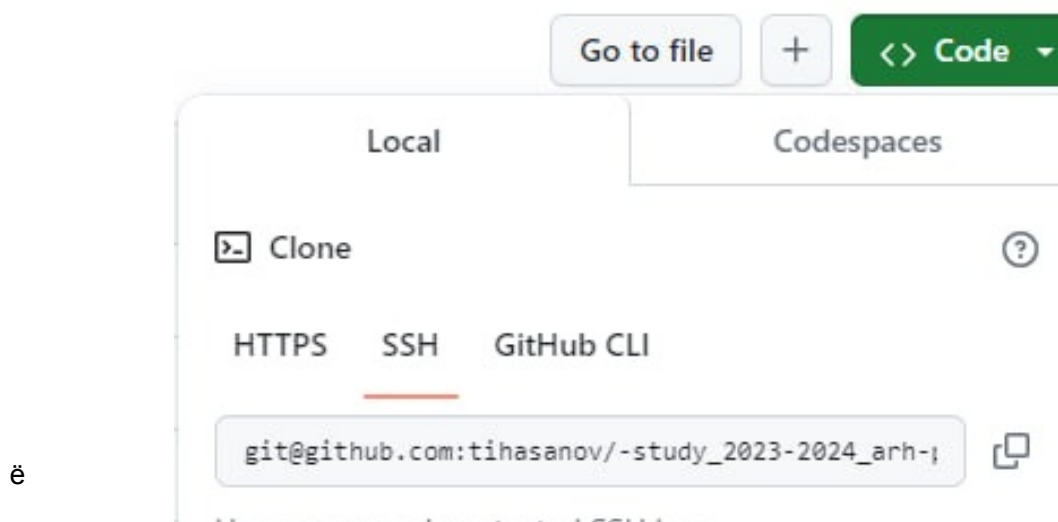


Рис. 4.19: Окно с ссылкой для копирования репозитория

4.6 Настройка каталога курса

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

```
[tihasanov@tihasanov Архитектура компьютера]$ cd arch-pc  
[tihasanov@tihasanov arch-pc]$
```

Рис. 4.20: Перемещение между директориями

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

```
[tihasanov@tihasanov arch-pc]$ rm package.json  
[tihasanov@tihasanov arch-pc]$
```

Рис. 4.21: Удаление файлов

Создаю необходимые каталоги (рис. 4.22).

```
[tihasanov@tihasanov arch-pc]$ echo arch-pc > COURSE  
[tihasanov@tihasanov 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).

```
[tihasanov@tihasanov arch-pc]$ git add .
warning: You ran 'git add' with neither '-A (--all)' or '--ignore-removal',
whose behaviour will change in Git 2.0 with respect to paths you removed.
Paths like 'package.json' that are
removed from your working tree are ignored with this version of Git.

* 'git add --ignore-removal <pathspec>', which is the current default,
  ignores paths you removed from your working tree.

* 'git add --all <pathspec>' will let you also record the removals.

Run 'git status' to check the paths you removed from your working tree.
[tihasanov@tihasanov arch-pc]$ git commit -am "Добавление новых файлов и и
изменений"
[master d3c5754] Добавление новых файлов и изменений
 2 files changed, 1 insertion(+), 14 deletions(-)
 delete mode 100644 package.json
```

Рис. 4.23: Добавление и сохранение изменений на сервере

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

```
[tihasanov@tihasanov arch-pc]$ git push
warning: push.default is unset; its implicit value is changing in
Git 2.0 from 'matching' to 'simple'. To squelch this message
and maintain the current behavior after the default changes, use:

  git config --global push.default matching

To squelch this message and adopt the new behavior now, use:

  git config --global push.default simple

See 'git help config' and search for 'push.default' for further informatio
n.
(the 'simple' mode was introduced in Git 1.7.11. Use the similar mode
'current' instead of 'simple' if you sometimes use older versions of Git)

Counting objects: 5, done.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 325 bytes | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote: To git@github.com:tihasanov/-study_2023-2024_arh-pc.git
       f88c5c4..d3c5754 master -> master
[tihasanov@tihasanov arch-pc]$
```

Рис. 4.24: Выгрузка изменений на сервер

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

-study_2023-2024_arh-pc / labs /

tihasanov Добавление новых файлов и изменени1 be257c1 - 1 minute ago History

Name	Last commit message	Last commit date
..		
lab01	Добавление новых файлов и изменени1	1 minute ago
lab02	Добавление новых файлов и изменени1	1 minute ago
lab03	Добавление новых файлов и изменени1	1 minute ago
lab04	Добавление новых файлов и изменени1	1 minute ago
lab05	Добавление новых файлов и изменени1	1 minute ago
lab06	Добавление новых файлов и изменени1	1 minute ago
lab07	Добавление новых файлов и изменени1	1 minute ago
lab08	Добавление новых файлов и изменени1	1 minute ago
lab09	Добавление новых файлов и изменени1	1 minute ago
lab10	Добавление новых файлов и изменени1	1 minute 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).

```
[tihasanov@tihasanov report]$ touch Л02_Хасанов_отчет
[tihasanov@tihasanov report]$
```

Рис. 4.26: Создание файла

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

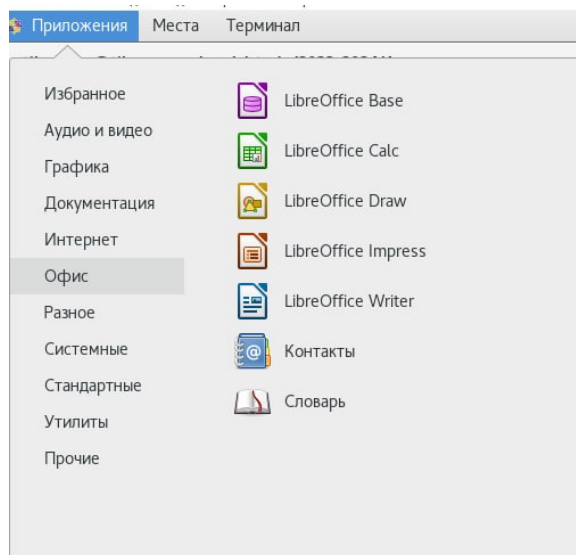


Рис. 4.27: Меню приложений

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

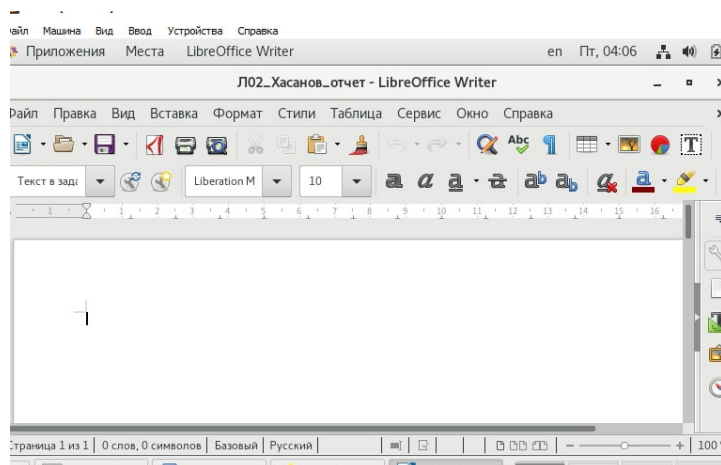


Рис. 4.28: Работа с отчетом в текстовом процессоре

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

```
tihasanov@tihasanov report]$ cd ../../lab01/report
tihasanov@tihasanov 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).

```
[tihasanov@tihasanov Загрузки]$ ls
Л01_Хасанов_отчет.pdf  Л01_Хасанов_отчет.pdf
[tihasanov@tihasanov Загрузки]$ █
```

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).

```
е [tihasanov@tihasanov report]$ cp ~/Загрузки/Л01_Хасанов_отчет.pdf .
[tihasanov@tihasanov report]$ ls
lib image Makefile pandoc report.md Л01_Хасанов_отчет.pdf
[tihasanov@tihasanov report]$ █
```

Рис. 4.31: Копирование файла

Добавляю файл `Л01_Хасанов_отчет` (рис. 4.32).

```
tihasanov@tihasanov report]$ git add "Л01_Хасанов_отчет.pdf"
tihasanov@tihasanov report]$ █
```

Рис. 4.32: Добавление файла на сервер

То же самое делаю для отчета по второй лабораторной работе: перехожу в директорию labs/lab02/report с помощью cd, добавляю с помощью git add нужный файл, сохраняю изменения с помощью git commit (рис. 4.33).

```
[tihasanov@tihasanov labs]$ cd lab02/report
[tihasanov@tihasanov report]$ git add Л02_Хасанов_отчет
[tihasanov@tihasanov report]$ git commit -m "Add existing file"
[master 8171e5e] Add existing file
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 labs/lab02/report/Л02_Хасанов_отчет
[tihasanov@tihasanov report]$
```

Рис. 4.33: Подкаталоги и файлы в репозитории

Отправляю в центральный репозиторий сохраненные изменения командой git push -f origin master (рис. 4.34).

```
[tihasanov@tihasanov report]$ git push -f origin master
Counting objects: 15, done.
Compressing objects: 100% (11/11), done.
Writing objects: 100% (11/11), 1.70 KiB | 0 bytes/s, done.
Total 11 (delta 5), reused 0 (delta 0)
remote: Resolving deltas: 100% (5/5), completed with 2 local objects.
remote: To git@github.com:tihasanov/-study_2023-2024_arh-pc.git
be257c1..8171e5e master -> master
[tihasanov@tihasanov report]$
```

Рис. 4.34: Отправка в центральный репозиторий сохраненных изменений

Проверяю на сайте GitHub правильность выполнения заданий. Вижу, что пояснение к совершенным действиям отображается (рис. 4.35).

lab01	Добавление Л01_Хасанов_отчет.pdf	18 minutes ago
lab02	Add existing file	2 minutes ago

Рис. 4.35: Страница каталога в репозитории

При просмотре изменений так же вижу, что были добавлены файлы с отчетами по лабораторным работам (рис. 4.36).

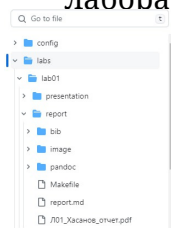


Рис. 4.36: Добавление файла на сервер

Вижу, что отчеты по лабораторным работам находятся в соответствующих каталогах репозитория: отчет по первой - в lab01/report (рис. 4.37), по второй – в lab02/report (рис. 4.38).

Name	Last commit message	Last commit date
..		
bib	Добавление новых файлов и изменени1	2 hours ago
image	Добавление новых файлов и изменени1	2 hours ago
pandoc	Добавление новых файлов и изменени1	2 hours ago
Makefile	Добавление новых файлов и изменени1	2 hours ago
report.md	Добавление новых файлов и изменени1	2 hours ago
л01_Хасанов_отчет.pdf	Добавление л01_Хасанов_отчет.pdf	20 minutes ago

Рис. 4.37: Каталог lab01/report

Name	Last commit message	Last commit date
..		
bib	Добавление новых файлов и изменени1	2 hours ago
image	Добавление новых файлов и изменени1	2 hours ago
pandoc	Добавление новых файлов и изменени1	2 hours ago
Makefile	Добавление новых файлов и изменени1	2 hours ago
report.md	Добавление новых файлов и изменени1	2 hours ago
л02_Хасанов_отчет	Add existing file	4 minutes ago

Рис. 4.38: Каталог lab02/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.

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.