

Отчёт по лабораторной работе №2

Дисциплина: архитектура компьютера

Гомазкова Алина

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	11
4.3	Создание SSH-ключа	12
4.4	Создание рабочего пространства и репозитория курса на основе шаблона	14
4.5	Создание репозитория курса на основе шаблона	14
4.6	Настройка каталога курса	18
5	Выводы	21
	Список литературы	22

Список иллюстраций

4.1	Заполнение данных учетной записи GitHub	10
4.2	Аккаунт GitHub	11
4.3	Предварительная конфигурация git	11
4.4	Настройка кодировки	11
4.5	Создание имени для начальной ветки	12
4.6	Параметр autocrlf	12
4.7	Параметр safecrlf	12
4.8	Установка утилиты xclip	13
4.9	Копирование содержимого файла	13
4.10	Окно SSH and GPG keys.Добавление ключа	14
4.11	Создание рабочего пространства	14
4.12	Страница шаблона для репозитория	15
4.13	Окно создания репозитория	16
4.14	Созданный репозиторий	16
4.15	Перемещение между директориями	17
4.16	Клонирование репозитория	17
4.17	Окно с ссылкой для копирования репозитория	17
4.18	Перемещение между директориями	18
4.19	Удаление файлов	18
4.20	Создание каталогов	18
4.21	Добавление и сохранение изменений на сервере	19
4.22	Выгрузка изменений на сервер	19
4.23	Страница репозитория	20

Список таблиц

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

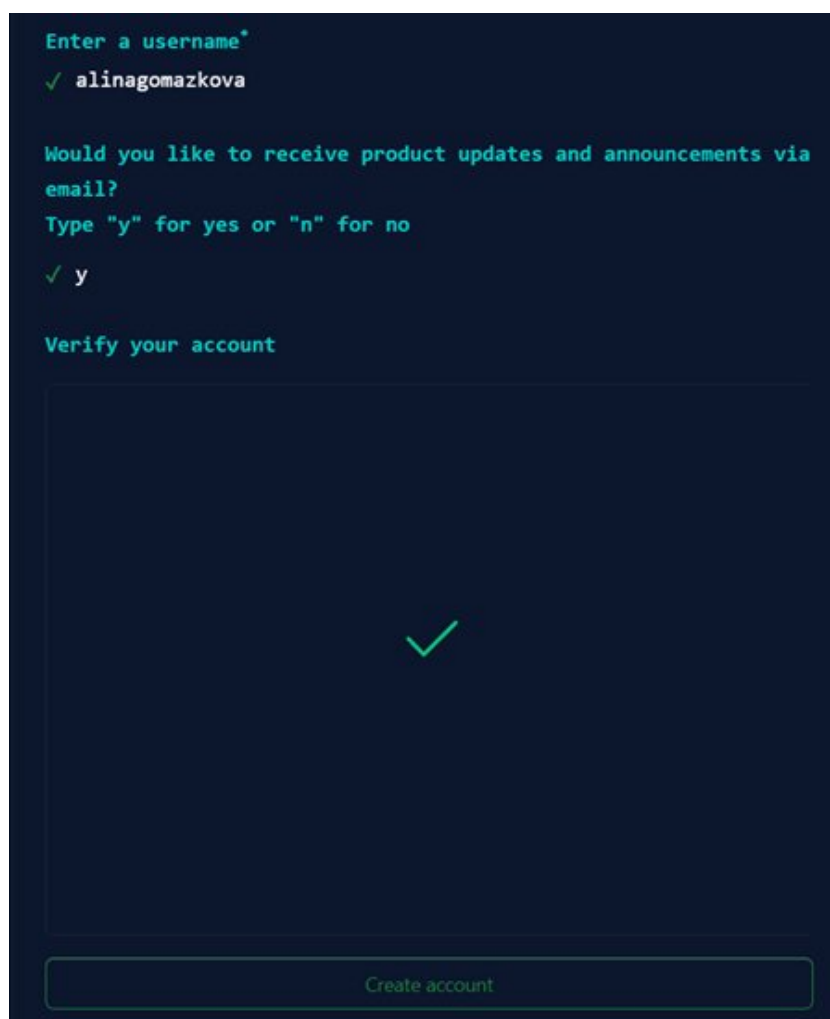
Системы контроля версий (Version Control System, VCS) применяются при работе нескольких человек над одним проектом. Обычно основное дерево проекта хранится в локальном или удалённом репозитории, к которому настроен доступ для участников проекта. При внесении изменений в содержание проекта система контроля 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. Далее я заполнила основные данные учетной записи (рис.[4.1]).



Enter a username*

✓ alinagomazkova

Would you like to receive product updates and announcements via email?

Type "y" for yes or "n" for no

✓ y

Verify your account

✓

Create account

Рис. 4.1: Заполнение данных учетной записи GitHub

Аккаунт создан (рис. [4.2]).

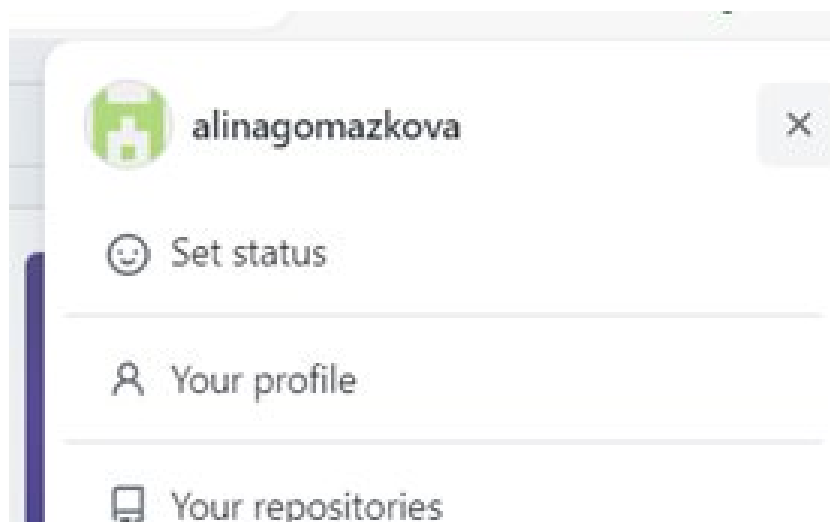


Рис. 4.2: Аккаунт GitHub

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]).

```
[alinagomazkova@fedora ~]$ git config --global user.name "<Alina Gomazkova>"
[alinagomazkova@fedora ~]$ git config --global user.email "<"1032235008@pfur.ru">"
```

Рис. 4.3: Предварительная конфигурация git

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

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

Рис. 4.4: Настройка кодировки

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

```
[alinagomazkova@fedora ~]$ 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. CR и LF – это символы, которые можно использовать для обозначения разрыва строки в текстовых файлах (рис.[4.6]).

```
[alinagomazkova@fedora ~]$ git config --global core.autocrlf input
```

Рис. 4.6: Параметр autocrlf

Задаю параметр safecrlf со значением warn, так Git будет проверять преобразование на обратимость. При значении warn Git только выведет предупреждение, но будет принимать необратимые конвертации (рис.[4.7]).

```
[alinagomazkova@fedora ~]$ 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. Ключ автоматически сохранится в каталоге `~/.ssh/` (рис.[4.3]).

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 отмени суперпользователя, введя в начале команды sudo (рис.[4.8]).

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

* Ожидание в очереди...
* Загрузка списка пакетов...
Следующие пакеты должны быть установлены:
xclip-0.13-19.git11c6a61.fc38.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...

[alinagomazkova@10 ~]$
```

Рис. 4.8: Установка утилиты xclip

Копирую открытый ключ из директории, в которой он был сохранен, с помощью утилиты xclip (рис.[4.9]).

```
[alinagomazkova@10 ~]$ cat ~/.ssh/id_rsa.pub | xclip -sel clip
[alinagomazkova@10 ~]$
```

Рис. 4.9: Копирование содержимого файла

Открываю браузер, захожу на сайт GitHub. Открываю свой профиль и выбираю страницу «SSH and GPG keys». Нажимаю кнопку «New SSH key». Вставляю скопированный ключ в поле «Key». В поле Title указываю имя для ключа. Нажимаю «Add SSH-key», чтобы завершить добавление ключа (рис.[4.10]).

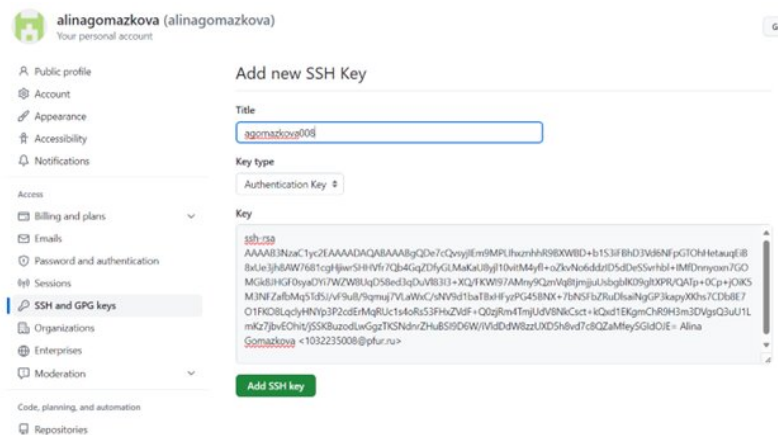


Рис. 4.10: Окно SSH and GPG keys.Добавление ключа

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.[4.11]).

```
[alinagomazkova@10 ~]$ mkdir -p work/study/2023-2024/"Архитектура компьютера"
[alinagomazkova@10 ~]$ ls
test  Видео  Загрузки  Музыка  'Рабочий стол'
work  Документы  Изображения  Общедоступные  Шаблоны
[alinagomazkova@10 ~]$
```

Рис. 4.11: 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.[4.12]).

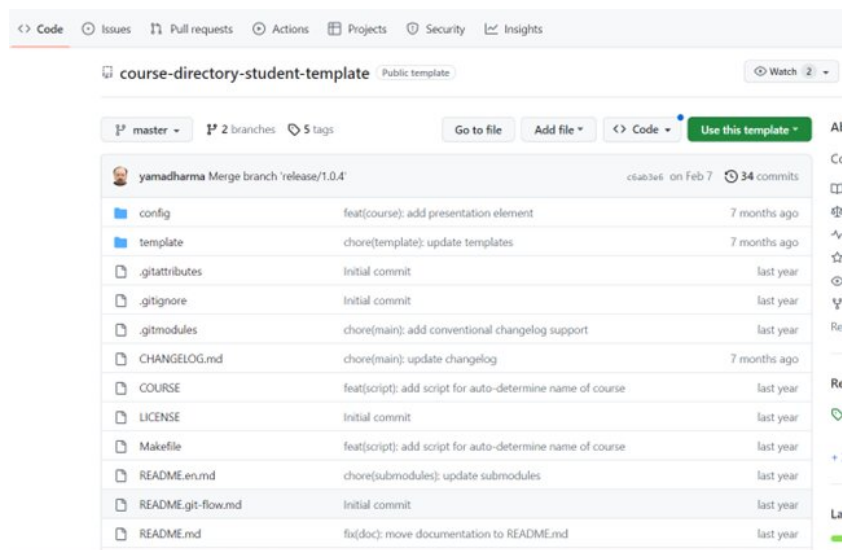


Рис. 4.12: Страница шаблона для репозитория

В открывшемся окне задаю имя репозитория (Repository name): study_2023–2024_arh-рс и создаю репозиторий, нажимаю на кнопку «Create repository from template» (рис.[4.13]).

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 * alinagomazkova / Repository name * study_2023-2024_arh-pc
study_2023-2024_arh-pc is available.

Great repository names are short and memorable. Need inspiration? How about [super-duper-pancake](#) ?

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. 4.13: Окно создания репозитория

Репозиторий создан (рис.[4.14]).

alinagomazkova Initial commit	#286224 1 minute ago	🕒 1 commit
config	Initial commit	1 minute ago
template	Initial commit	1 minute ago
.gitattributes	Initial commit	1 minute ago
.gitignore	Initial commit	1 minute ago
.gitmodules	Initial commit	1 minute ago
CHANGELOG.md	Initial commit	1 minute ago
COURSE	Initial commit	1 minute ago
LICENSE	Initial commit	1 minute ago
Makefile	Initial commit	1 minute ago
README.en.md	Initial commit	1 minute ago
README.git-flow.md	Initial commit	1 minute ago
README.md	Initial commit	1 minute ago

Рис. 4.14: Созданный репозиторий

Через терминал перехожу в созданный каталог курса с помощью утилиты `cd` (рис.[4.15]).


```
[alinagomazkova@10 ~]$ cd ~/work/study/2023-2024/'Архитектура компьютера'
[alinagomazkova@10 Архитектура компьютера]$
```

Рис. 4.15: Перемещение между директориями

Клонирую созданный репозиторий с помощью команды `git clone --recursive git@github.com:/study_2023-2024_arh-pc.git arch-pc` (рис.[4.16]).

```
[alinagomazkova@10 Архитектура компьютера]$ git clone --recursive git@github.com
:alinagomazkova/study_2023-2024_arh-pc.git arch-pc
Клонирование в «arch-pc»...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvC0qU.
This key is not known by any other names.
```

Рис. 4.16: Клонирование репозитория

Копирую ссылку для клонирования на странице созданного репозитория, сначала перейдя в окно «code», далее выбрав в окне вкладку «SSH» (рис.[4.17]).

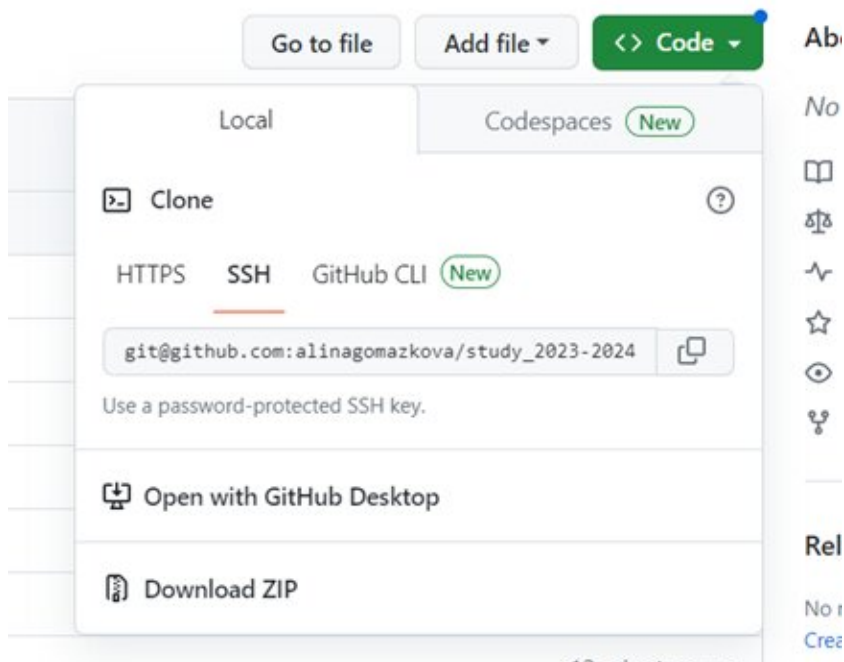


Рис. 4.17: Окно с ссылкой для копирования репозитория

4.6 Настройка каталога курса

Перехожу в каталог arch-pc с помощью утилиты cd (рис.[4.18]).

```
301es02  
[alinagomazkova@10 Архитектура компьютера]$ cd ~/work/study/2023-2024/Архитектур  
a\ компьютера/arch-pc  
[alinagomazkova@10 arch-pc]$
```

Рис. 4.18: Перемещение между директориями

Удаляю лишние файлы с помощью утилиты rm (рис.[4.19]).

```
[alinagomazkova@10 arch-pc]$ rm package.json  
[alinagomazkova@10 arch-pc]$
```

Рис. 4.19: Удаление файлов

Создаю необходимые каталоги (рис.[4.20]).

```
[alinagomazkova@10 arch-pc]$ rm package.json  
[alinagomazkova@10 arch-pc]$ echo arch-pc > COURSE  
[alinagomazkova@10 arch-pc]$ make  
[alinagomazkova@10 arch-pc]$
```

Рис. 4.20: 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.22.)(рис. 4.21).

```

[alinagomazkova@10 arch-pc]$ git add .
[alinagomazkova@10 arch-pc]$ git commit -am 'feat(main): make course structure'
[master 534d700] feat(main): make course structure
199 files changed, 54725 insertions(+), 14 deletions(-)
create mode 100644 labs/README.md
create mode 100644 labs/README.ru.md
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/Makefile

```

Рис. 4.21: Добавление и сохранение изменений на сервере

Отправляю все на сервер с помощью push (рис.[4.22]).

```

[alinagomazkova@10 arch-pc]$ git push
Перечисление объектов: 37, готово.
Подсчет объектов: 100% (37/37), готово.
Сжатие объектов: 100% (29/29), готово.
Запись объектов: 100% (35/35), 342.14 КиБ | 957.00 КиБ/с, готово.
Всего 35 (изменений 4), повторно использовано 0 (изменений 0), повторно использо
вано пакетов 0
remote: Resolving deltas: 100% (4/4), completed with 1 local object.
To github.com:alinagomazkova/study_2023-2024_arh-pc.git
 f286224..534d700 master -> master
[alinagomazkova@10 arch-pc]$

```

Рис. 4.22: Выгрузка изменений на сервер

Проверяю правильность выполнения работы сначала на самом сайте GitHub (рис.[4.23]).

Alina Gomaškova feat(main): make course structure		5344700 · 5 minutes ago	History
Name	Last commit message	Last commit date	
..			
lab01	feat(main): make course structure	5 minutes ago	
lab02	feat(main): make course structure	5 minutes ago	
lab03	feat(main): make course structure	5 minutes ago	
lab04	feat(main): make course structure	5 minutes ago	
lab05	feat(main): make course structure	5 minutes ago	
lab06	feat(main): make course structure	5 minutes ago	
lab07	feat(main): make course structure	5 minutes ago	
lab08	feat(main): make course structure	5 minutes ago	
lab09	feat(main): make course structure	5 minutes ago	
lab10	feat(main): make course structure	5 minutes ago	
lab11	feat(main): make course structure	5 minutes ago	

Рис. 4.23: Страница репозитория

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.

Список литературы

1.Архитектура ЭВМ