Шаблон отчёта по лабораторной работе

Простейший вариант

Дельгадильо Валерия

Содержание

# 1 Цель работы

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

# 2 Теоретическое введение

## 2.1 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.

## 2.2 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.

## 2.3 Основные команды git

Наиболее часто используемые команды git представлены в таблице 1

Таблица 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 изменения |
| git add | добавить все изменённые и/или созданные файлы и/или каталоги |
| git add  имена\_фаилов | добавить конкретные изменённые и/или созданные файлы и/или каталоги |
| git rm  имена\_фаилов | 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 Лабораторной работы

## 3.1 Настройка github

Для выполнения лабораторных работ предлагается использовать Github. Создайте учётную запись на сайте https://github.com/ и заполните основные данные.

Созданная учетная запись показана на рисунках [[1](#fig:3.1)] и [[2](#fig:3.2)].

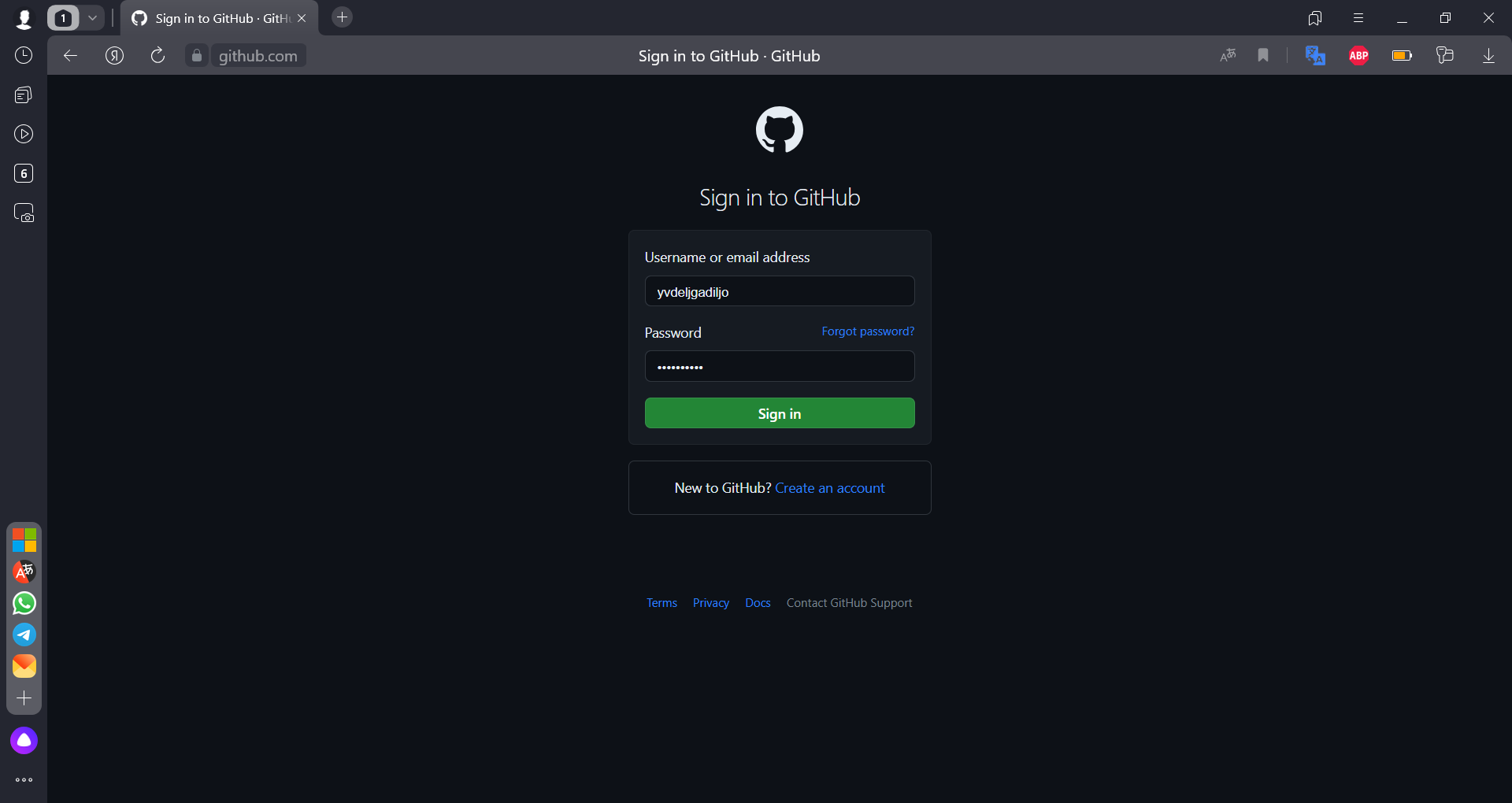


Figure 1:

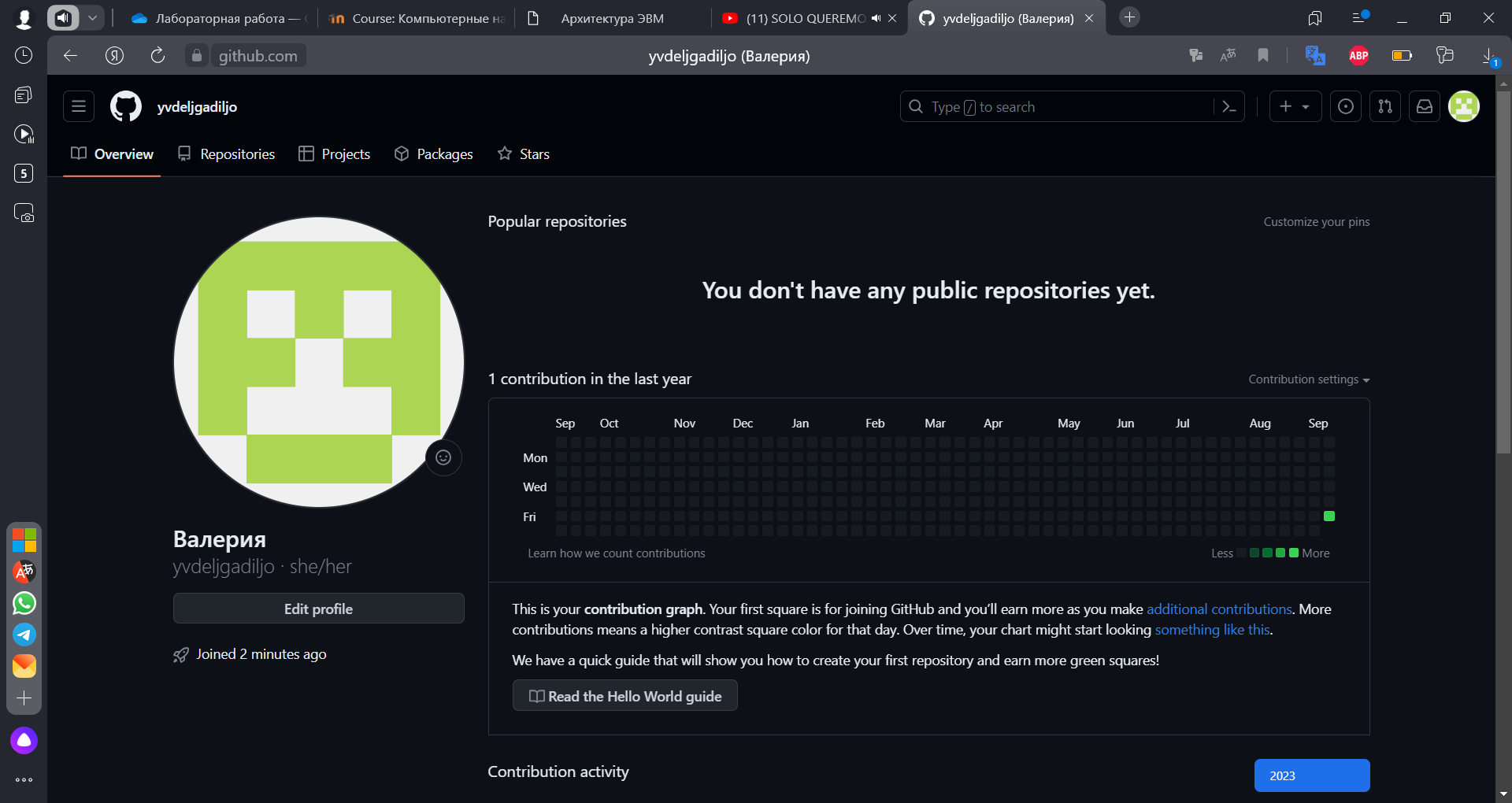


Figure 2:

## 3.2 Базовая настройка git

Сначала сделаем предварительную конфигурацию git.

1. Откройте терминал и введите следующие команды, указав имя и email владельца репозитория:

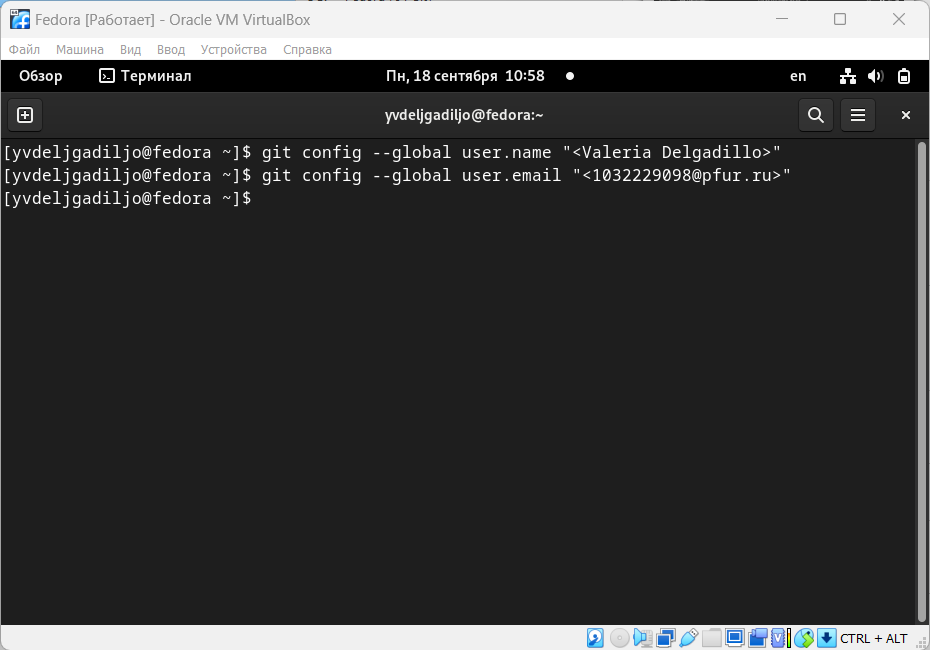


Figure 3:

1. Настроим utf-8 в выводе сообщений git и зададим имя начальной ветки (будем называть её master):

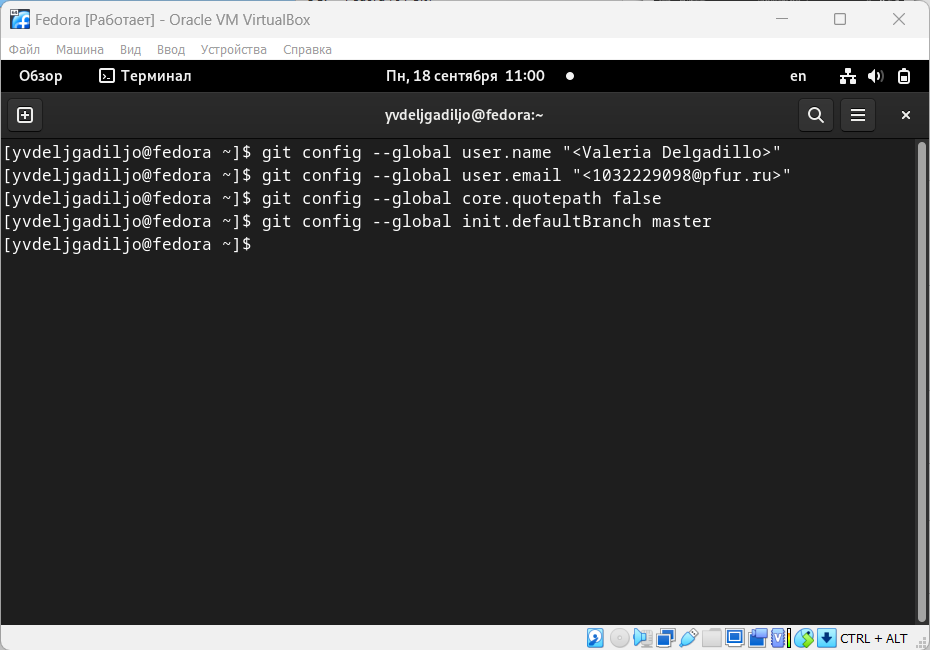


Figure 4:

1. Параметр autocrlf и параметр safecrlf:

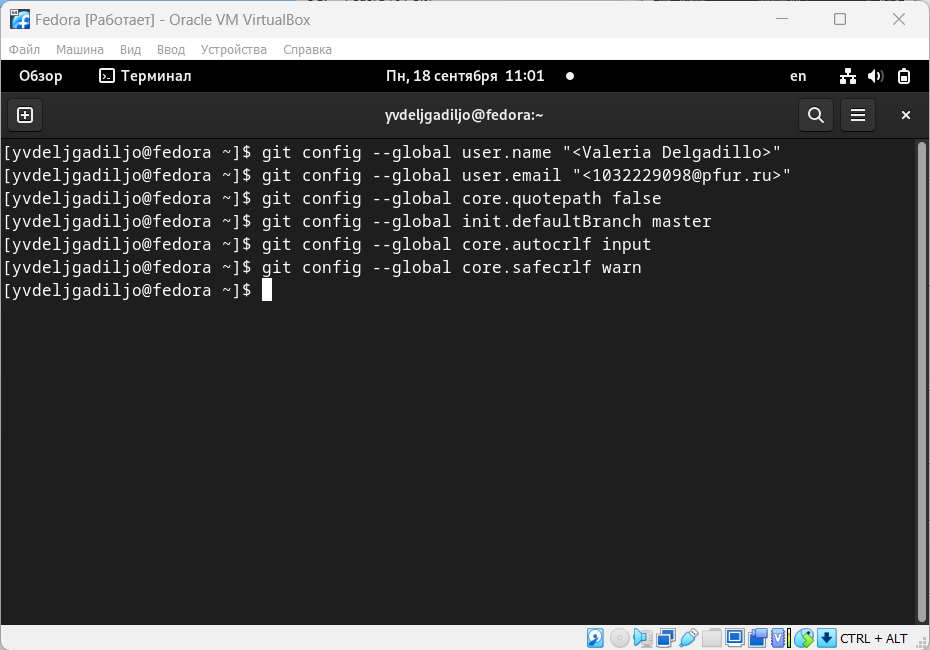


Figure 5:

## 3.3 Создание SSH ключа

Для последующей идентификации пользователя на сервере репозиториев необходимо сгенерировать пару ключей (приватный и открытый).

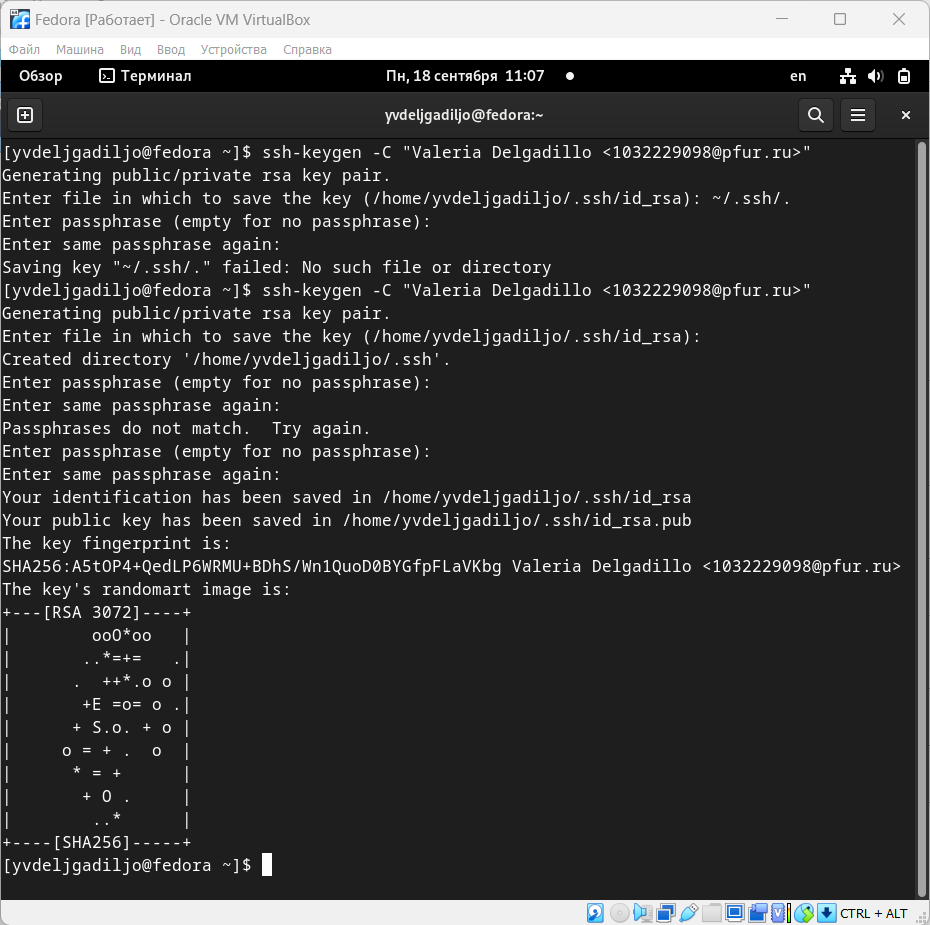


Figure 6:

Далее необходимо загрузить сгенерённый открытый ключ. Для этого зайти на сайт http://github.org/ под своей учётной записью и перейти в меню *Setting* . После этого выбрать в боковом меню *SSH and GPG* keys и нажать кнопку *New SSH key*. Скопировав из локальной консоли ключ в буфер обмена.

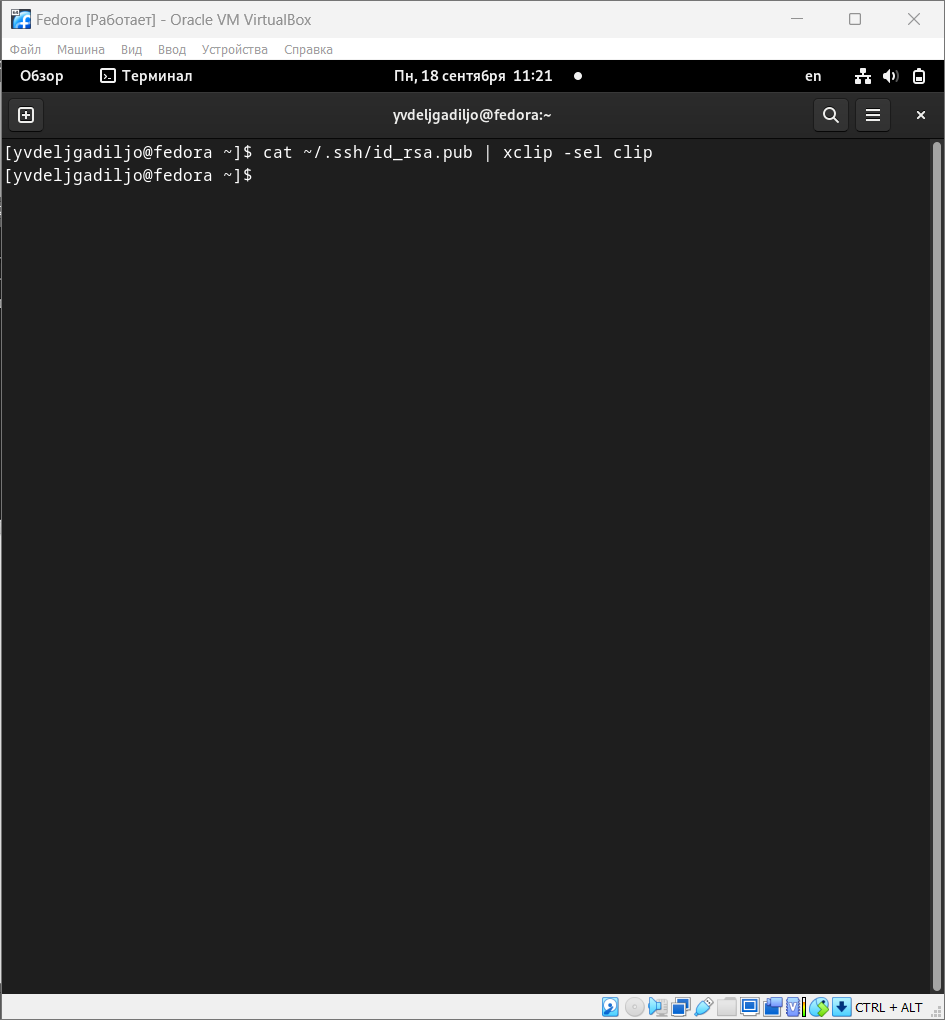


Figure 7:

Вставляем ключ в появившееся на сайте поле и указываем для ключа имя (Title).

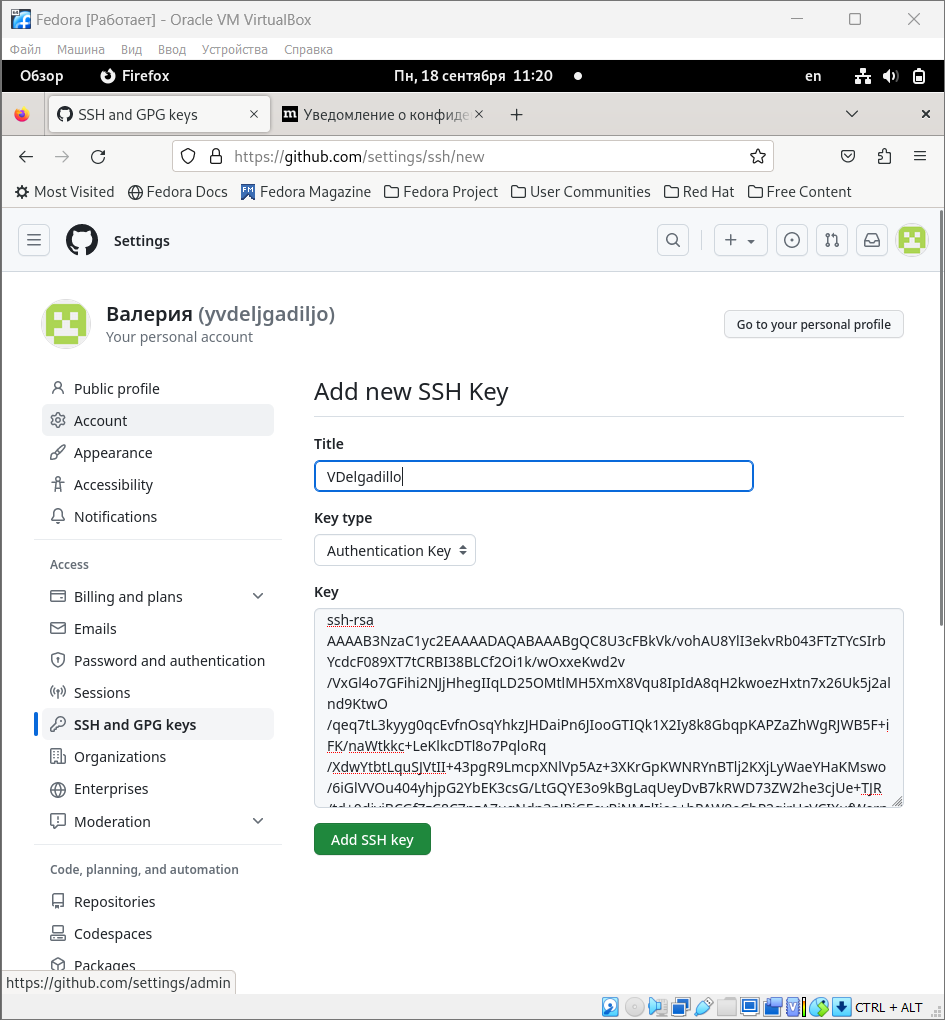


Figure 8:

## 3.4 Сознание рабочего пространства и репозитория курса на основе шаблона

Название проекта на хостинге git имеет вид: study\_<учебный год>\_<код предмета>. Откройте терминал и создайте каталог для предмета «Архитектура компьютера»:

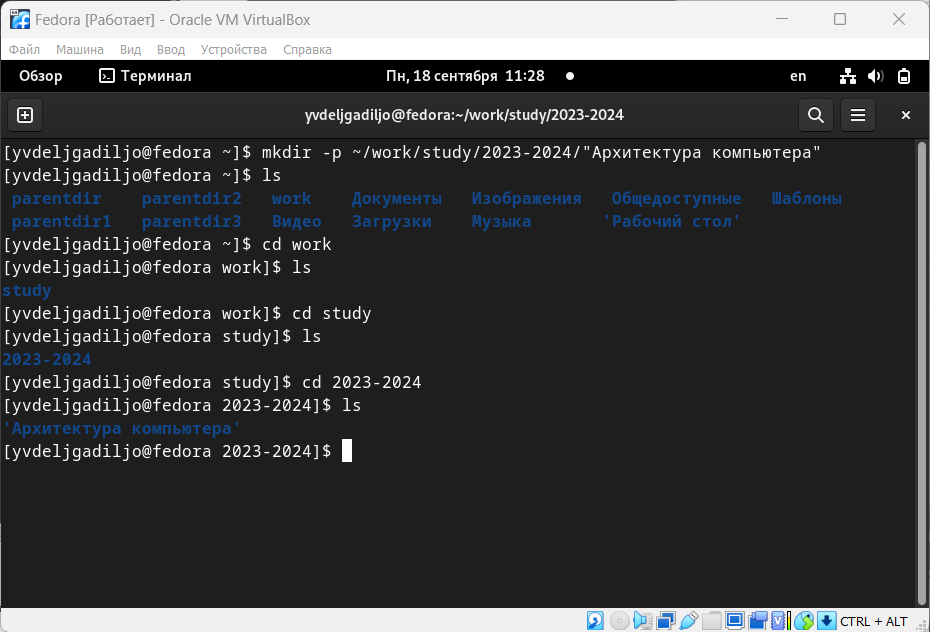


Figure 9:

## 3.5 Сознание репозитория курса на основе шаблона

Создайте репозиторий на основе шаблона через веб-интерфейс github. Репозиторий на основе шаблона можно создать через web-интерфейс github. Перейдите на станицу репозитория с шаблоном курса https://github.com/yamadharma/cour se-directory-student-template. Далее выберите Use this template.

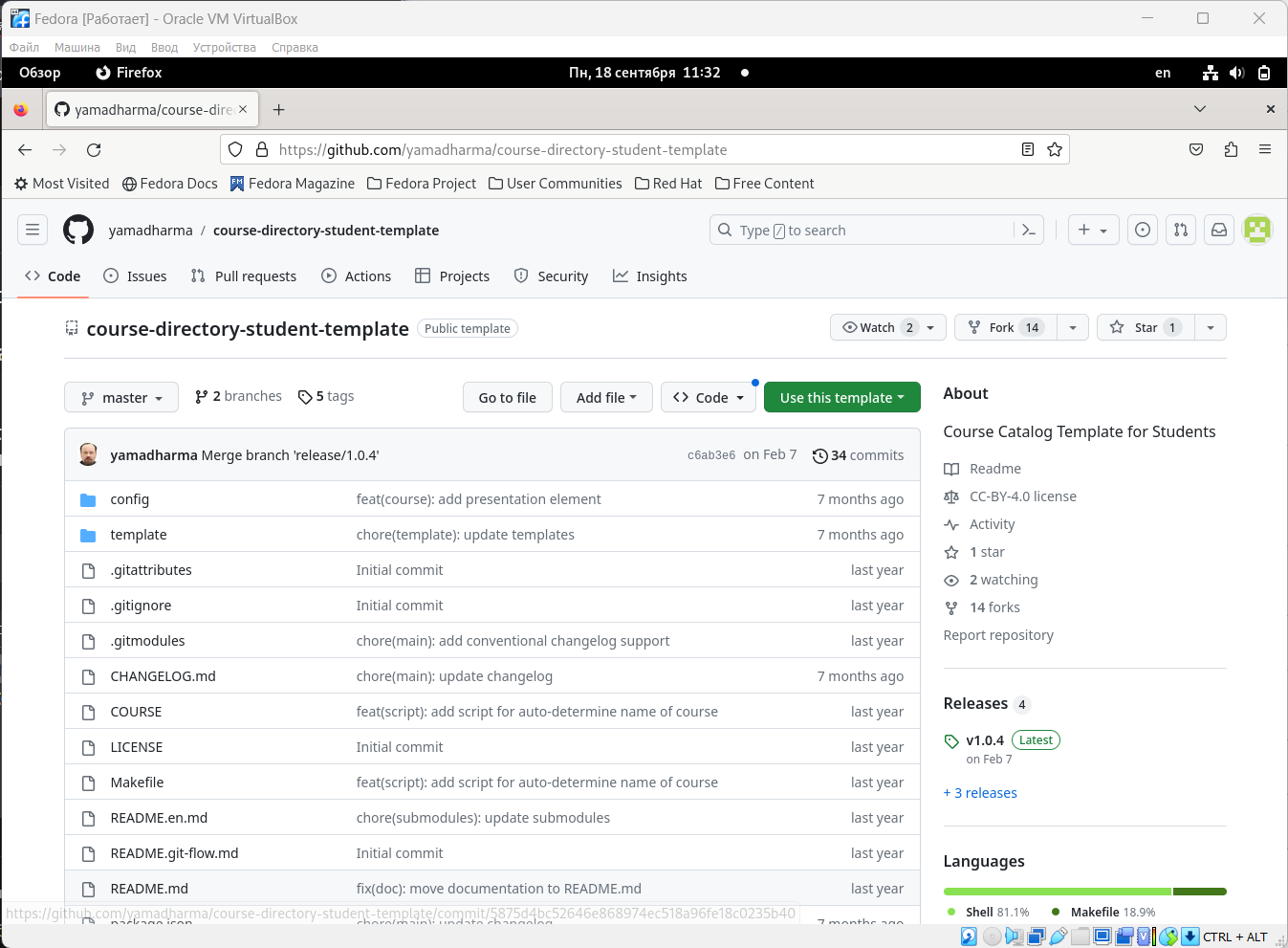


Figure 10:

В открывшемся окне задайте имя репозитория (Repository name) study\_2023–2024\_arhpc и создайте репозиторий (кнопка Create repository from template).

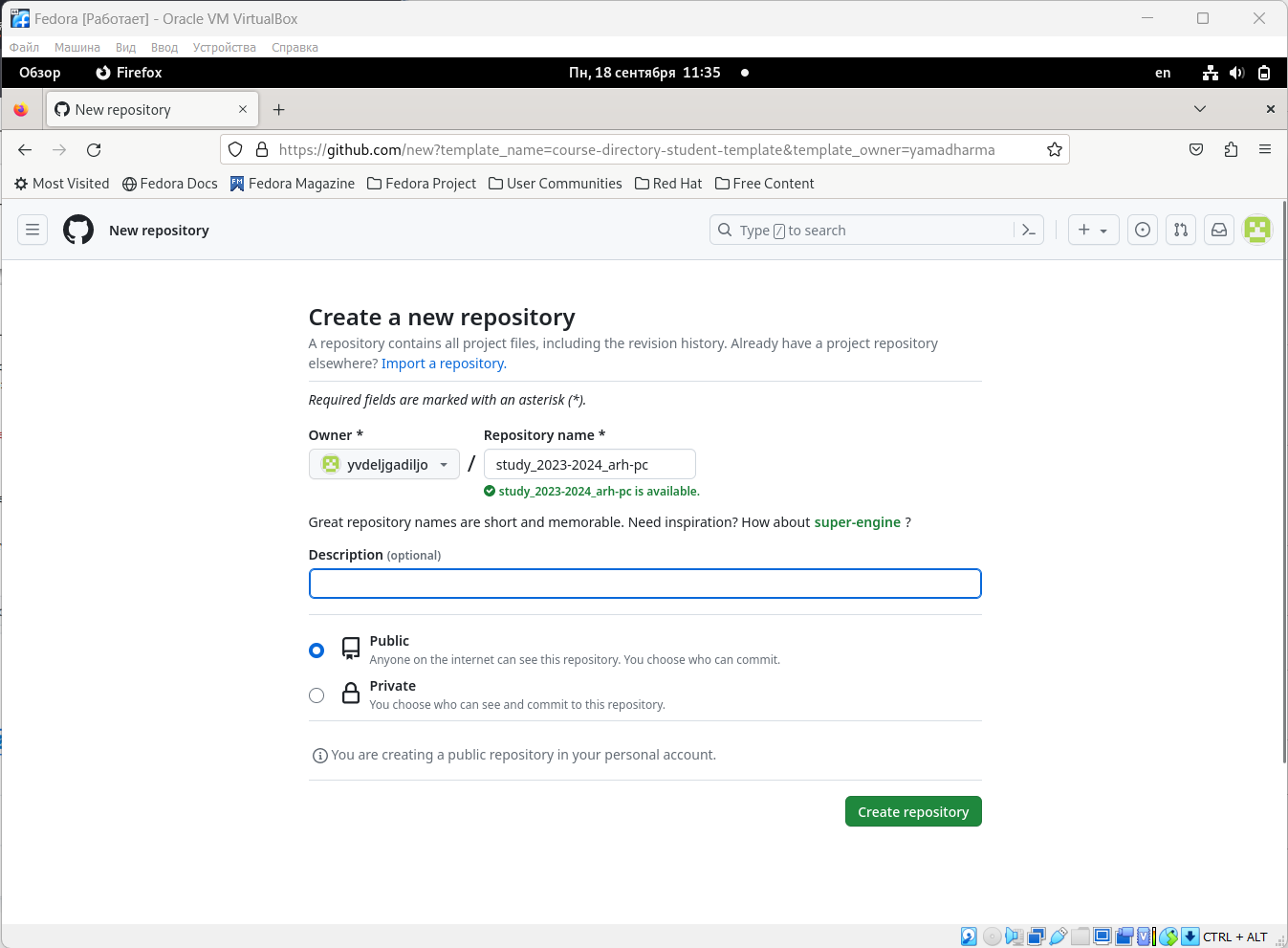


Figure 11:

Откройте терминал и перейдите в каталог курса и клонируйте созданный репозиторий.

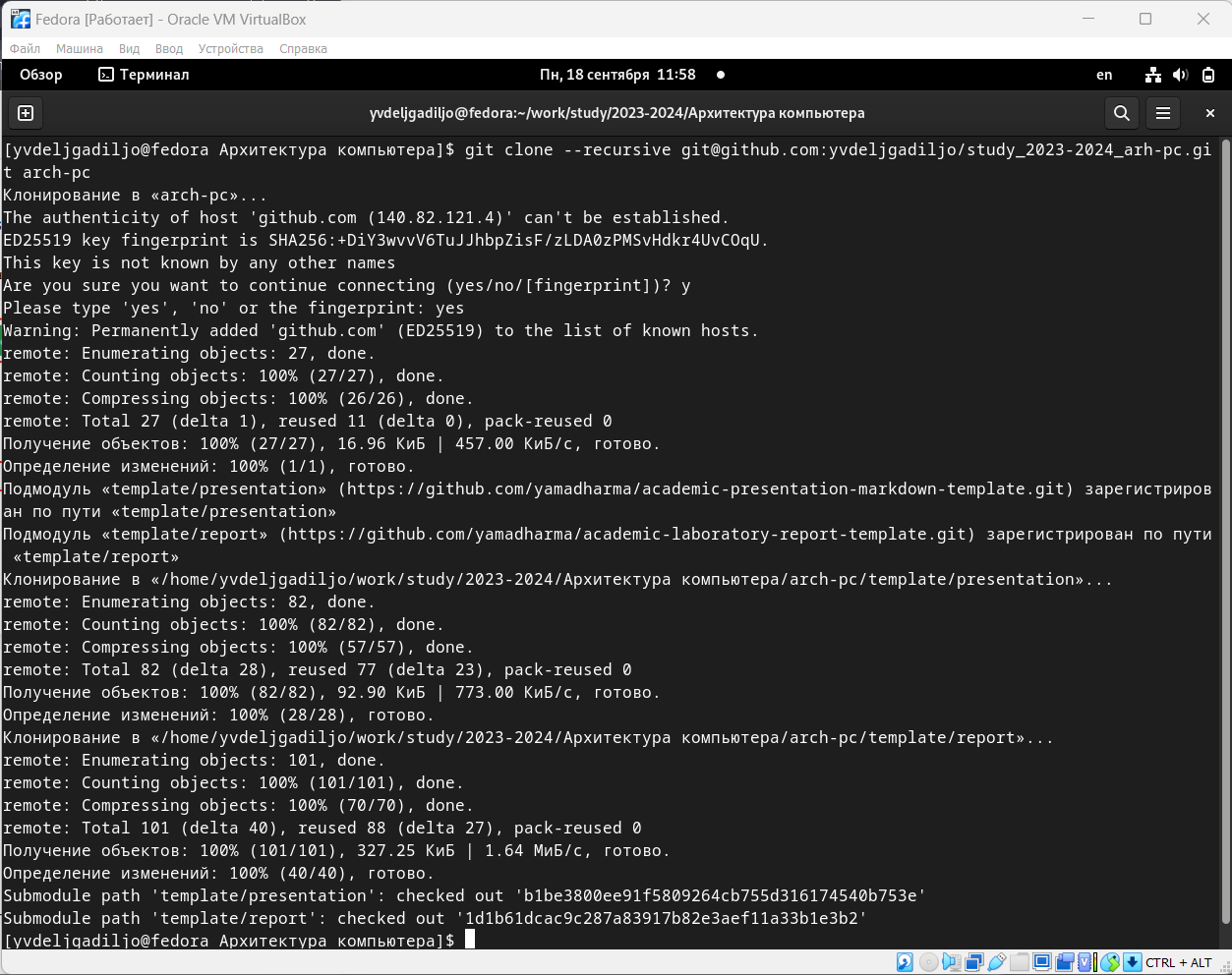


Figure 12:

## 3.6 Настройка каталога курса

Перейдите в каталог курса и удалите лишние файлы

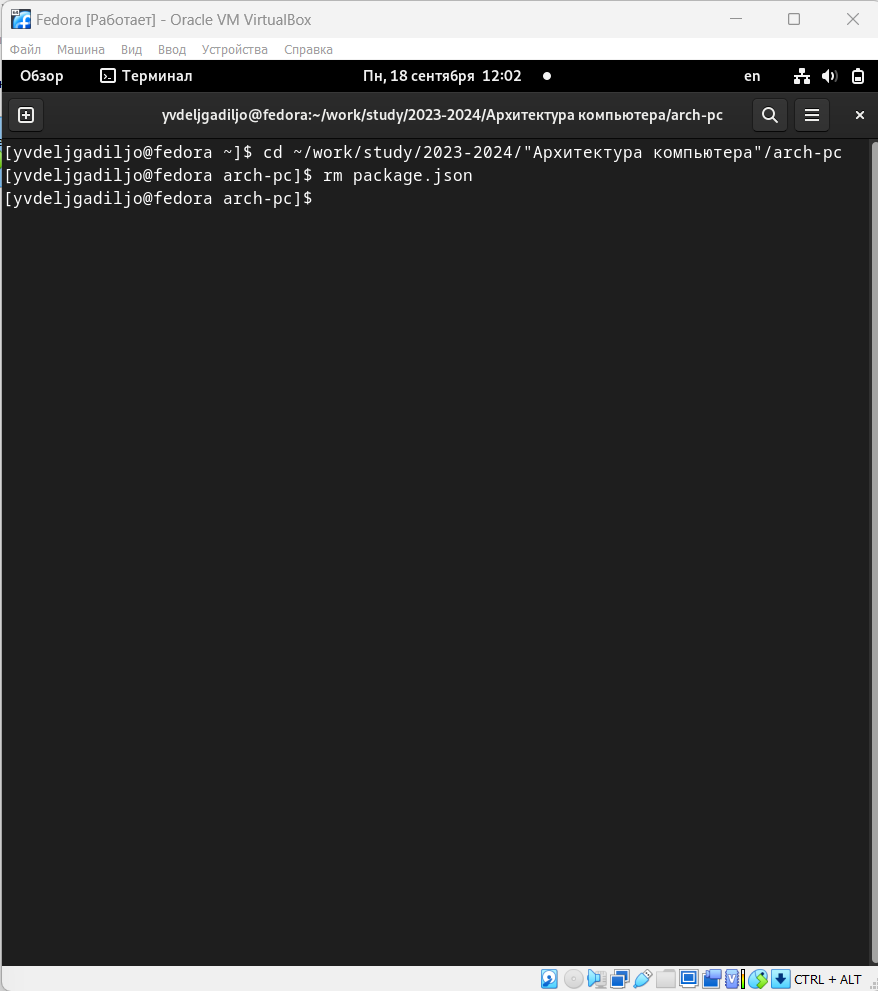


Figure 13:

Создайте необходимые каталоги:

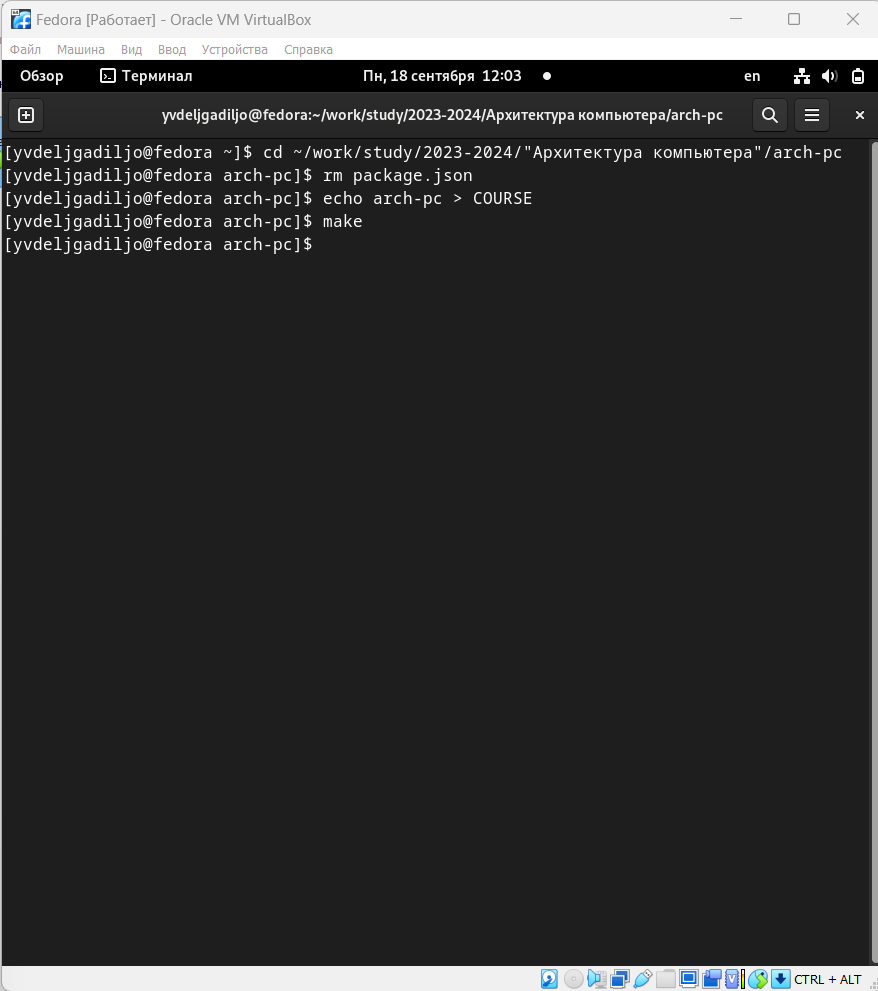


Figure 14:

Отправьте файлы на сервер:

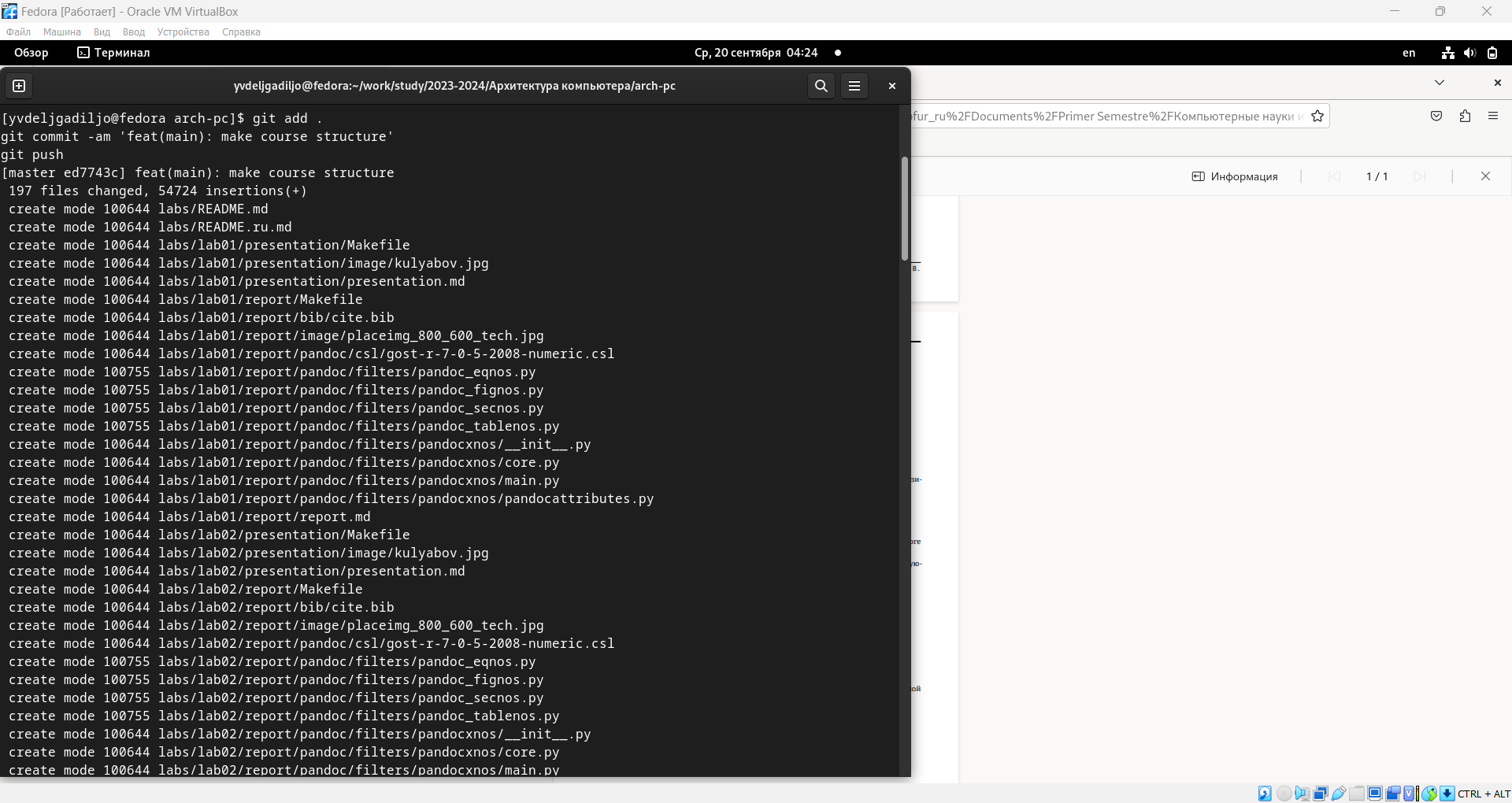


Figure 15:

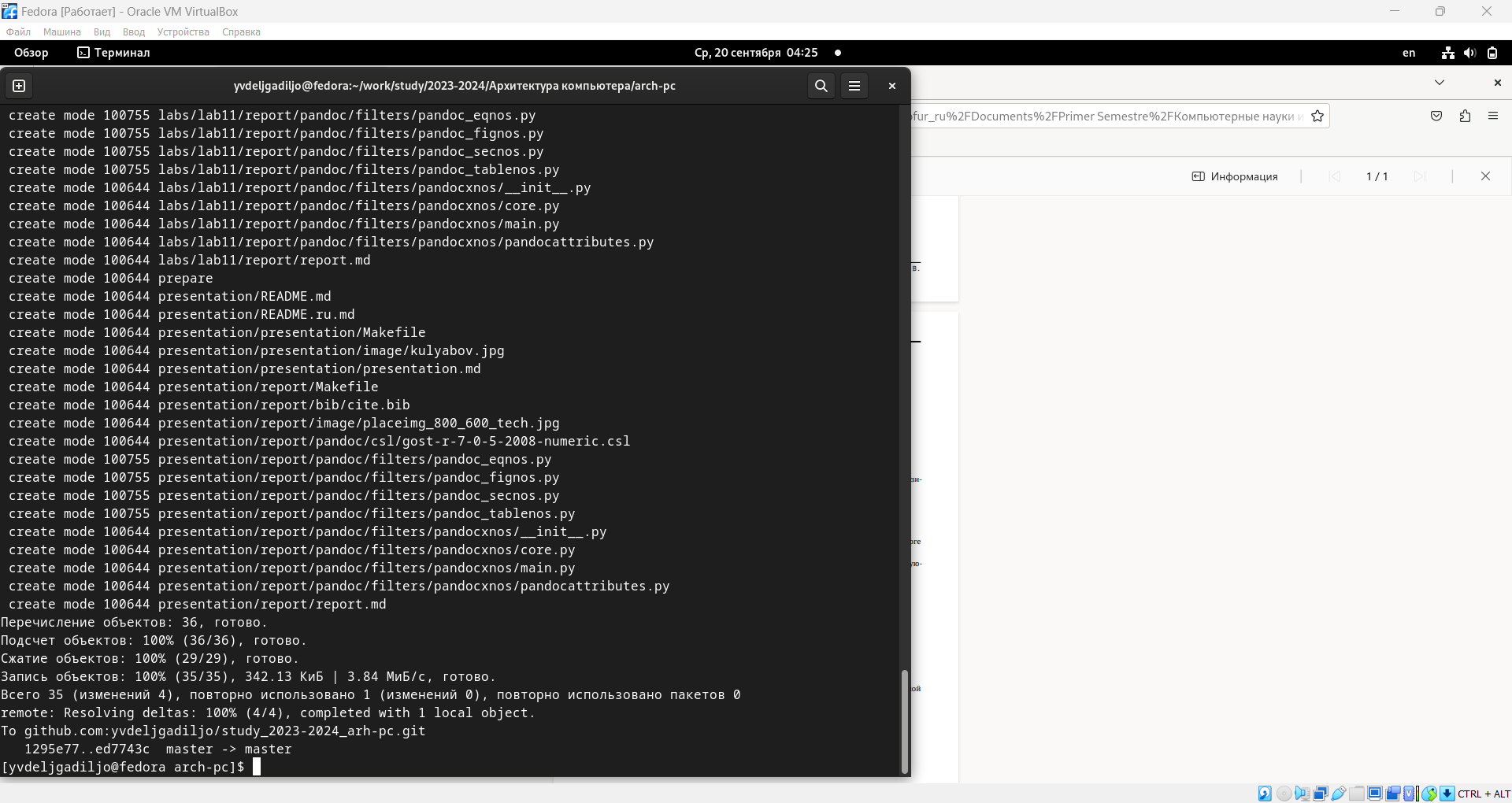


Figure 16:

# 4 Задание для самостоятельной работы

## 4.1 Создать лабораторный отчет в каталоге

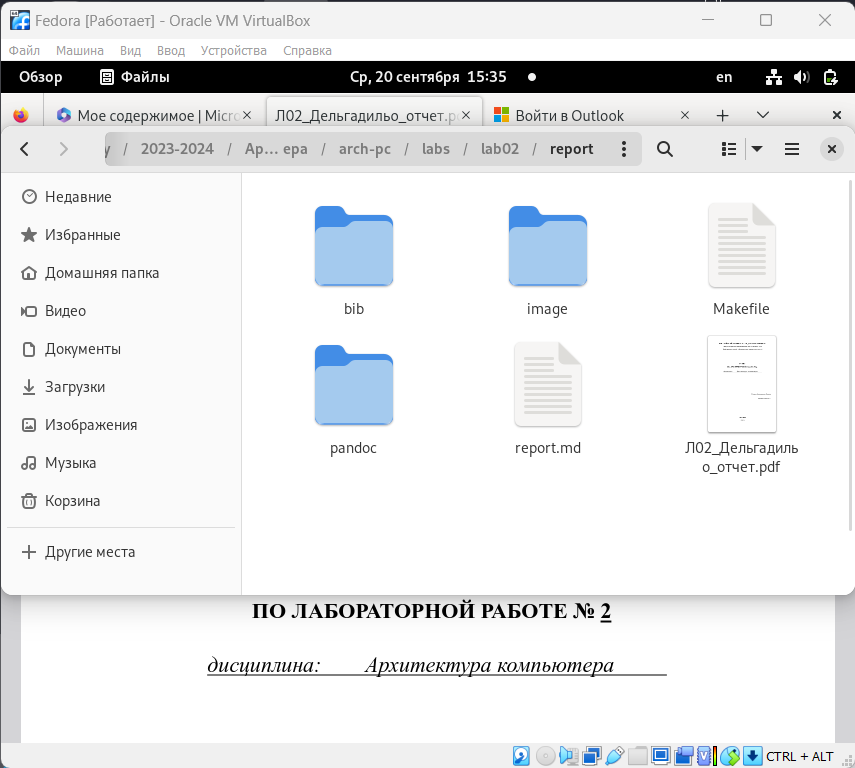


Figure 17:

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

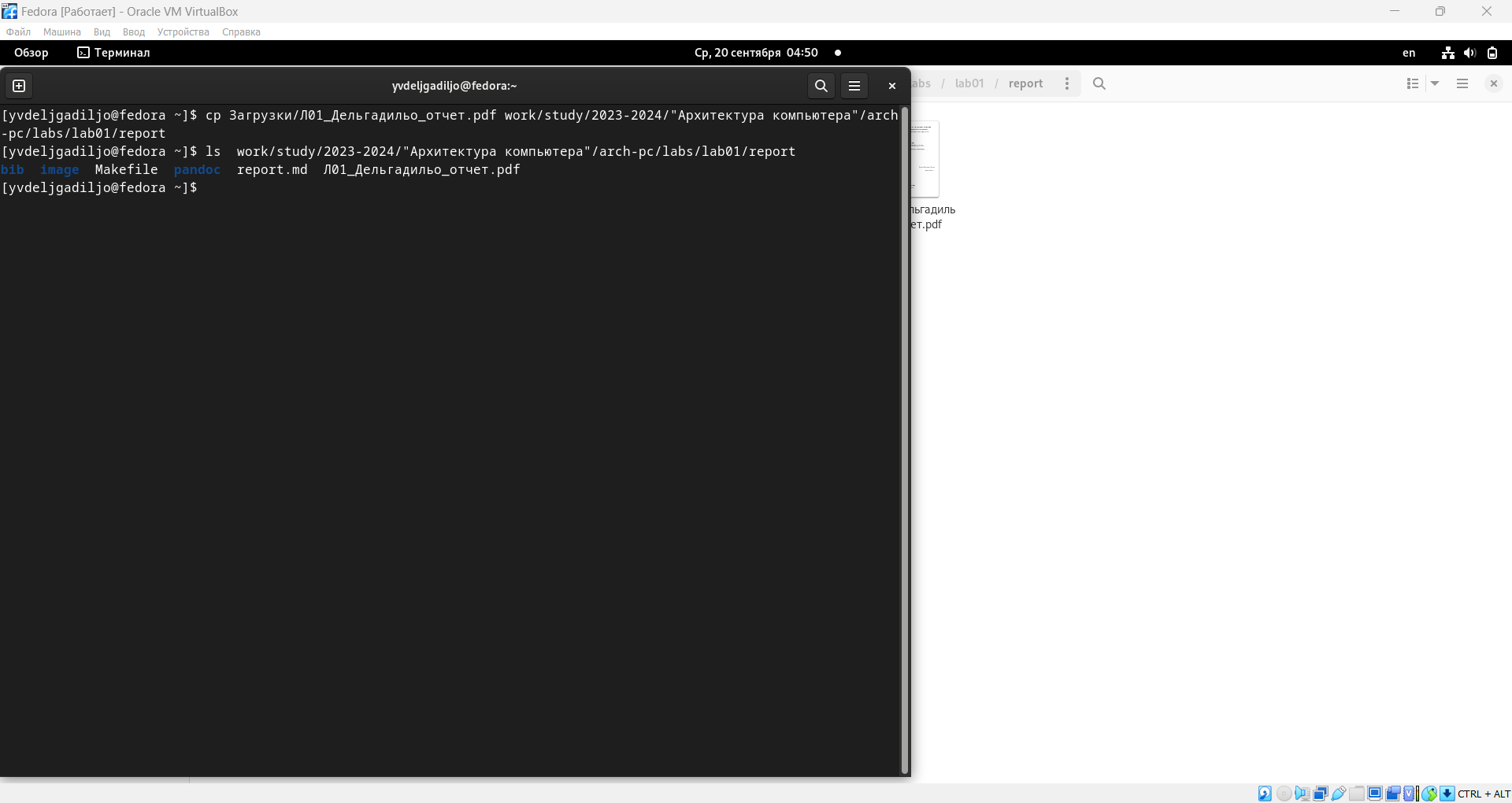


Figure 18:

## 4.3 Загрузите файлы на github

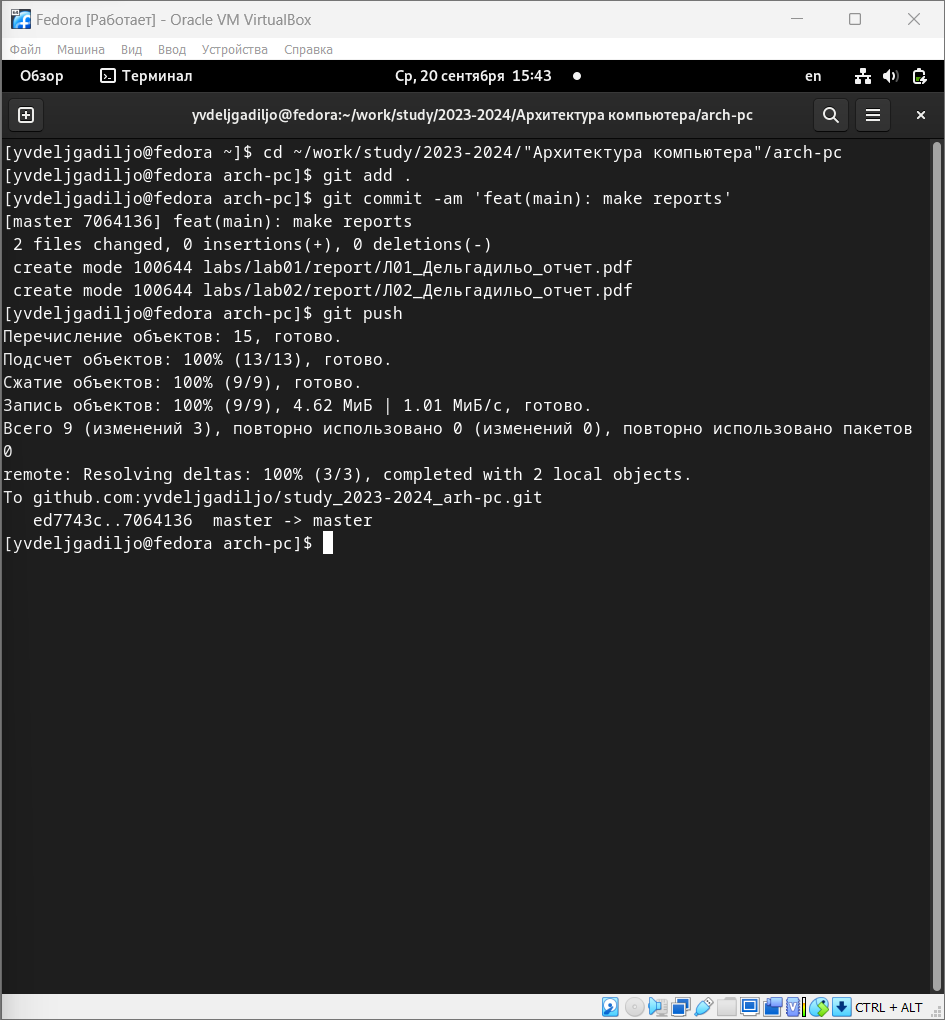


Figure 19:

# 5 Выводы

Благодаря этой лаборатории 2 вы узнали основные задачи, которые решает инструмент git и все его команды.

# 6 Список литературы

* GDB: The GNU Project Debugger. — URL: https://www.gnu.org/software/gdb/.
* GNU Bash Manual. — 2016. — URL: https://www.gnu.org/software/bash/manual/.
* Midnight Commander Development Center. — 2021. — URL: https://midnight-commander.org/.
* NASM Assembly Language Tutorials. — 2021. — URL: https://asmtutor.com/.
* *Newham* *C.* Learning the bash Shell: Unix Shell Programming. — O’Reilly Media, 2005. —354 с. — (In a Nutshell). — ISBN 0596009658. — URL: http://www.amazon.com/Learningbash-Shell-Programming-Nutshell/dp/0596009658.
* *Robbins A.* Bash Pocket Reference. — O’Reilly Media, 2016. — 156 с. — ISBN 978-1491941591.
* The NASM documentation. — 2021. — URL: https://www.nasm.us/docs.php.
* *Zarrelli G.* Mastering Bash. — Packt Publishing, 2017. — 502 с. — ISBN 9781784396879.
* *Колдаев В. Д., Лупин С. А.* Архитектура ЭВМ. — М. : Форум, 2018.
* *Куляс О. Л., Никитин К. А.* Курс программирования на ASSEMBLER. — М. : Солон-Пресс, 2017.
* *Новожилов О. П.* Архитектура ЭВМ и систем. — М. : Юрайт, 2016.
* Расширенный ассемблер: NASM. — 2021. — URL: https://www.opennet.ru/docs/RUS/nasm/.
* *Робачевский А., Немнюгин С., Стесик О.* Операционная система UNIX. — 2-е изд. — БХВПетербург, 2010. — 656 с. — ISBN 978-5-94157-538-1.
* *Столяров А.* Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm\_unix.
* *Таненбаум Э.* Архитектура компьютера. — 6-е изд. — СПб. : Питер, 2013. — 874 с. — (Классика Computer Science).
* *Таненбаум Э., Бос Х.* Современные операционные системы. — 4-е изд. — СПб. : Питер, 2015. — 1120 с. — (Классика Computer Science)