

# **Отчёт по лабораторной работе №2**

**дисциплина: Операционные системы**

Быстров Глеб Андреевич

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>3</b>
<b>2</b>	<b>Задание</b>	<b>4</b>
<b>3</b>	<b>Выполнение лабораторной работы</b>	<b>5</b>
<b>4</b>	<b>Контрольные вопросы</b>	<b>12</b>
<b>5</b>	<b>Выводы</b>	<b>16</b>

# 1 Цель работы

В данной лабораторной работе мне будет необходимо научиться оформлять отчёты с помощью легковесного языка разметки Markdown.

## 2 Задание

Сделать отчёт по предыдущей лабораторной работе в формате Markdown. В качестве отчёта предоставить отчёты в 3 форматах: pdf, docx и md.

### 3 Выполнение лабораторной работы

1. Создал учётную запись на платформе <https://github.com>. (рис. 3.1)

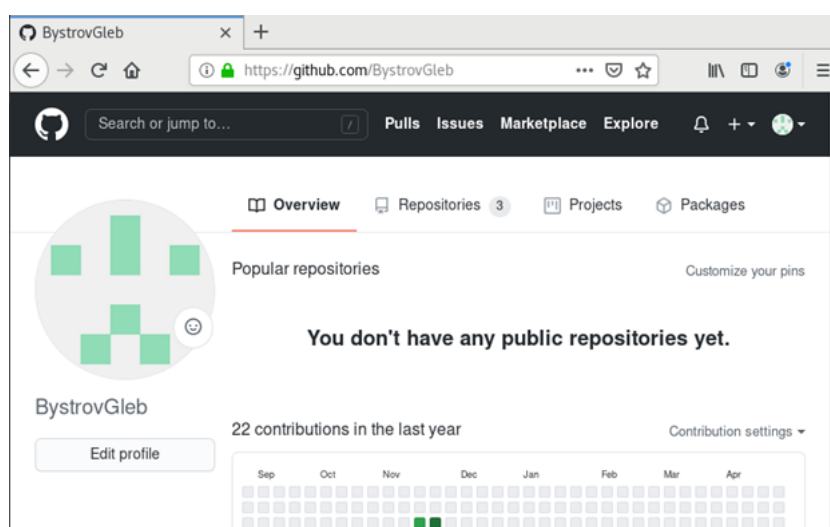


Figure 3.1: Учётная запись

2. Сгенерировал пару ключей (приватный и открытый). (рис. 3.2)



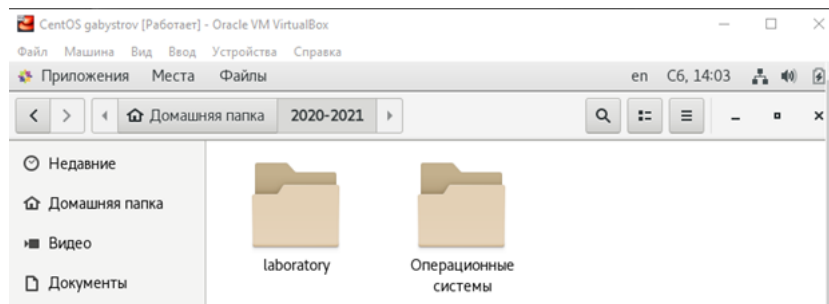


Figure 3.4: Структура каталога

5. Создал репозиторий на GitHub. (рис. 3.5)

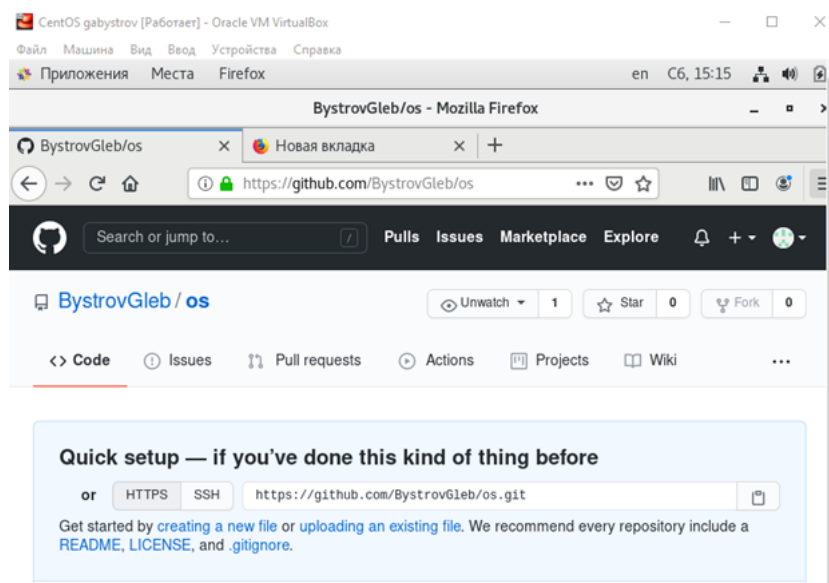


Figure 3.5: Репозиторий на GitHub

6. Подключил репозиторий к GitHub. (рис. 3.6)



Figure 3.6: Подключение репозитория

## 7. Добавил файл лицензии. (рис. 3.7)

```
[gabystrov@gabystrov laboratory]$ wget https://creativecommons.org/licenses/by/4.0/legalcode.txt -O LICENSE
--2021-05-01 16:12:36-- https://creativecommons.org/licenses/by/4.0/legalcode.txt
Распознаётся creativecommons.org (creativecommons.org)... 104.20.151.16, 172.67.34.140,
104.20.150.16, ...
Подключение к creativecommons.org (creativecommons.org)|104.20.151.16|:443... соединени
е установлено.
HTTP-запрос отправлен. Ожидание ответа... 200 OK
Длина: нет данных [text/plain]
Сохранение в: «LICENSE»

[ <=> ] 18 657 --.-K/s за 0,01s
2021-05-01 16:12:36 (1,26 MB/s) - «LICENSE» сохранён [18657]
[gabystrov@gabystrov laboratory]$
```

Figure 3.7: Добавление файла лицензии

## 8. Добавил шаблон игнорируемых файлов. (рис. 3.8)

```
[gabystrov@gabystrov laboratory]$ curl -L -s https://www.gitignore.io/api/list
1c,1c-bitrix,a-frame,actionscript,ada
adobe,advancedinstaller,adventuregamestudio,agda,al
alteraquartusii,altium,amplify,android,androidstudio
angular,anjuta,ansible,apachecordova,apachehadoop
appbuilder,appcelerator titanium,appcode,appcode+all,appcode+iml
appengine,aptanastudio,arcanist,archive,archives
archlinuxpackages,aspnetcore,assembler,ate,atmelstudio
ats,audio,automationstudio,autotools,autotools+strict
awr,azurefunctions,backup,ballerina,basercms
basic,batch,bazaar,bazel,bitrise
bitrix,bittorrent,blackbox,bloop,bluej
bookdown,bower,brickcc,buck,c
c++,cake,cakephp,cakephp2,cakephp3
calabash,carthage,certificates,ceylon,cfwheels
chefcookbook,chocolatey,clean,clion,clion+all
clion+iml,closure,cloud9,cmake,cocoapods
cocos2dx,cocoscreator,code,code-java,codeblocks
codecomposerstudio,codeigniter,codeio,codekit,codesniffer
coffeescript,commonlisp,compodoc,composer,compressed
compressedarchive,compression,conan,concrete5,coq
cordova,craftcms,crashlytics,crbasic,crossbar
crystal,cs-cart,csharp,cuda,cvs
cypressio,d,dart,darteditor,data
database,datarecovery,dbeaver,defold,delphi
```

Figure 3.8: Шаблон игнорируемых файлов

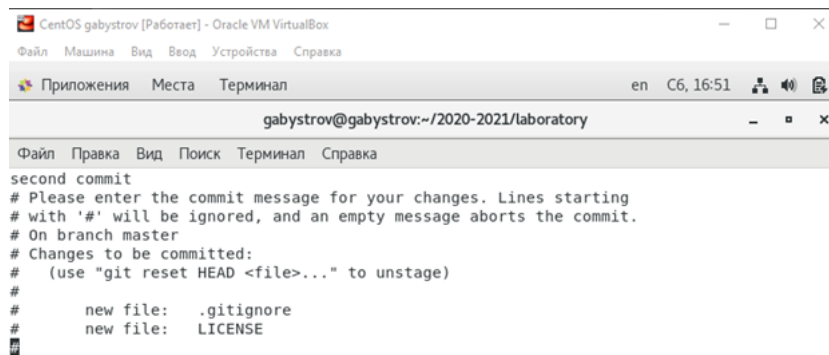
## 9. Скачал шаблон для C. (рис. 3.9)

```
zsh,zukencr8000[gabystrov@gabystrov laboratory]$ curl -L -s https://www.gitignore.io/ap
rec >> .gitigno
[gabystrov@gabystrov laboratory]$ curl -L -s https://www.gitignore.io/api/c >> .gitigno
re
[gabystrov@gabystrov laboratory]$
```

Figure 3.9: Шаблон для C

## 10. Добавил новые файлы. (рис. 3.10)

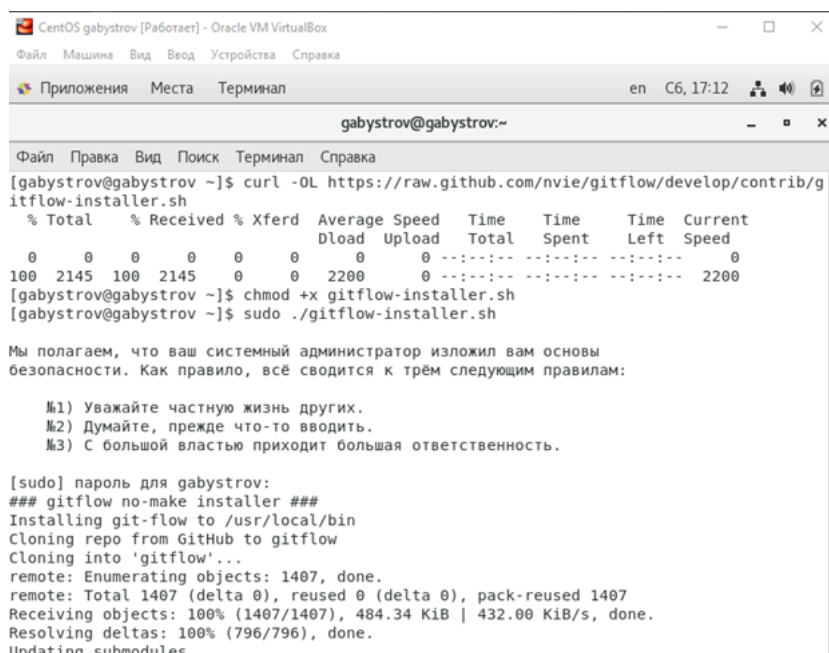




```
CentOS gabystrov [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
Приложения  Места  Терминал  en  C6, 16:51
gabystrov@gabystrov:~/2020-2021/laboratory
Файл  Правка  Вид  Поиск  Терминал  Справка
second commit
# Please enter the commit message for your changes. Lines starting
# with '#' will be ignored, and an empty message aborts the commit.
# On branch master
# Changes to be committed:
#   (use "git reset HEAD <file>..." to unstage)
#
#       new file:   .gitignore
#       new file:   LICENSE
```

Figure 3.10: Новые файлы

## 11. Скачал git-flow. (рис. 3.11)



```
CentOS gabystrov [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
Приложения  Места  Терминал  en  C6, 17:12
gabystrov@gabystrov:~
Файл  Правка  Вид  Поиск  Терминал  Справка
[gabystrov@gabystrov ~]$ curl -OL https://raw.githubusercontent.com/nvie/gitflow/develop/contrib/gitflow-installer.sh
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left   Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--    0
100 2145 100 2145    0     0 2200    0 --:--:-- --:--:-- --:--:-- 2200
[gabystrov@gabystrov ~]$ chmod +x gitflow-installer.sh
[gabystrov@gabystrov ~]$ sudo ./gitflow-installer.sh

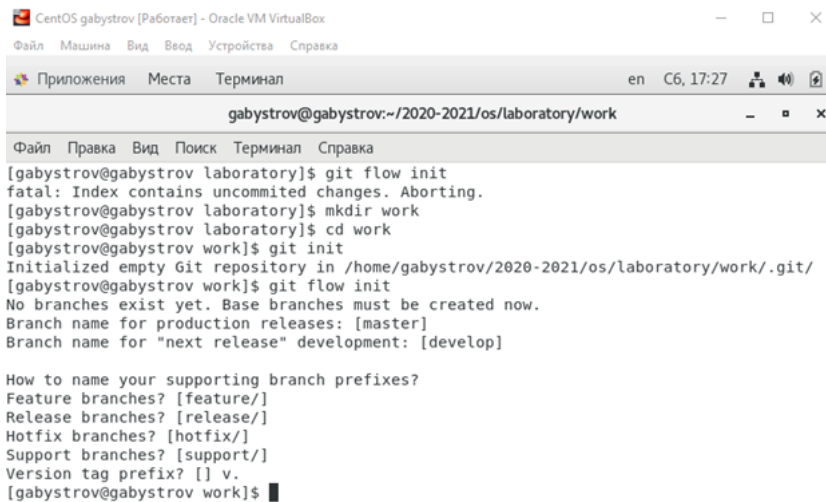
Мы полагаем, что ваш системный администратор изложил вам основы
безопасности. Как правило, всё сводится к трём следующим правилам:

    №1) Уважайте частную жизнь других.
    №2) Думайте, прежде что-то вводить.
    №3) С большой властью приходит большая ответственность.

[sudo] пароль для gabystrov:
### gitflow no-make installer ###
Installing git-flow to /usr/local/bin
Cloning repo from GitHub to gitflow
Cloning into 'gitflow'...
remote: Enumerating objects: 1407, done.
remote: Total 1407 (delta 0), reused 0 (delta 0), pack-reused 1407
Receiving objects: 100% (1407/1407), 484.34 KiB | 432.00 KiB/s, done.
Resolving deltas: 100% (796/796), done.
Updating submodules
```

Figure 3.11: git-flow

## 12. Провёл работу с конфигурацией git-flow. (рис. 3.12)

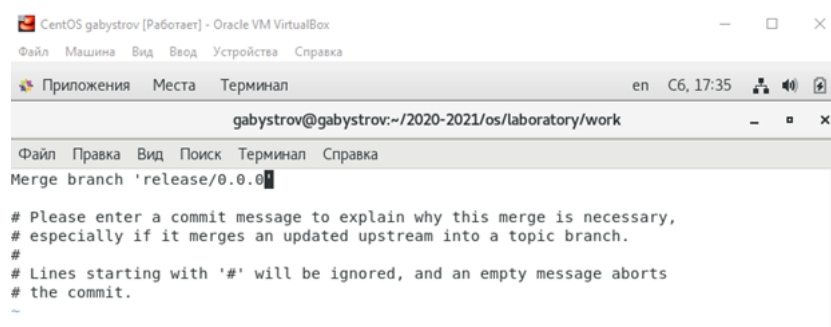


```
CentOS gabystrov [Работаer] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
Приложения  Места  Терминал  en  C6, 17:27
gabystrov@gabystrov:~/2020-2021/os/laboratory/work
Файл  Правка  Вид  Поиск  Терминал  Справка
[gabystrov@gabystrov laboratory]$ git flow init
fatal: Index contains uncommitted changes. Aborting.
[gabystrov@gabystrov laboratory]$ mkdir work
[gabystrov@gabystrov laboratory]$ cd work
[gabystrov@gabystrov work]$ git init
Initialized empty Git repository in /home/gabystrov/2020-2021/os/laboratory/work/.git/
[gabystrov@gabystrov work]$ git flow init
No branches exist yet. Base branches must be created now.
Branch name for production releases: [master]
Branch name for "next release" development: [develop]

How to name your supporting branch prefixes?
Feature branches? [feature/]
Release branches? [release/]
Hotfix branches? [hotfix/]
Support branches? [support/]
Version tag prefix? [ ] v.
[gabystrov@gabystrov work]$
```

Figure 3.12: Конфигурация git-flow

### 13. Создал релиз с версией. (рис. 3.13)



```
CentOS gabystrov [Работаer] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
Приложения  Места  Терминал  en  C6, 17:35
gabystrov@gabystrov:~/2020-2021/os/laboratory/work
Файл  Правка  Вид  Поиск  Терминал  Справка
Merge branch 'release/0.0.0'

# Please enter a commit message to explain why this merge is necessary,
# especially if it merges an updated upstream into a topic branch.
#
# Lines starting with '#' will be ignored, and an empty message aborts
# the commit.
~
~
~
```

Figure 3.13: Релиз с версией

### 14. Файлы успешно созданы и загружены. (рис. 3.14)

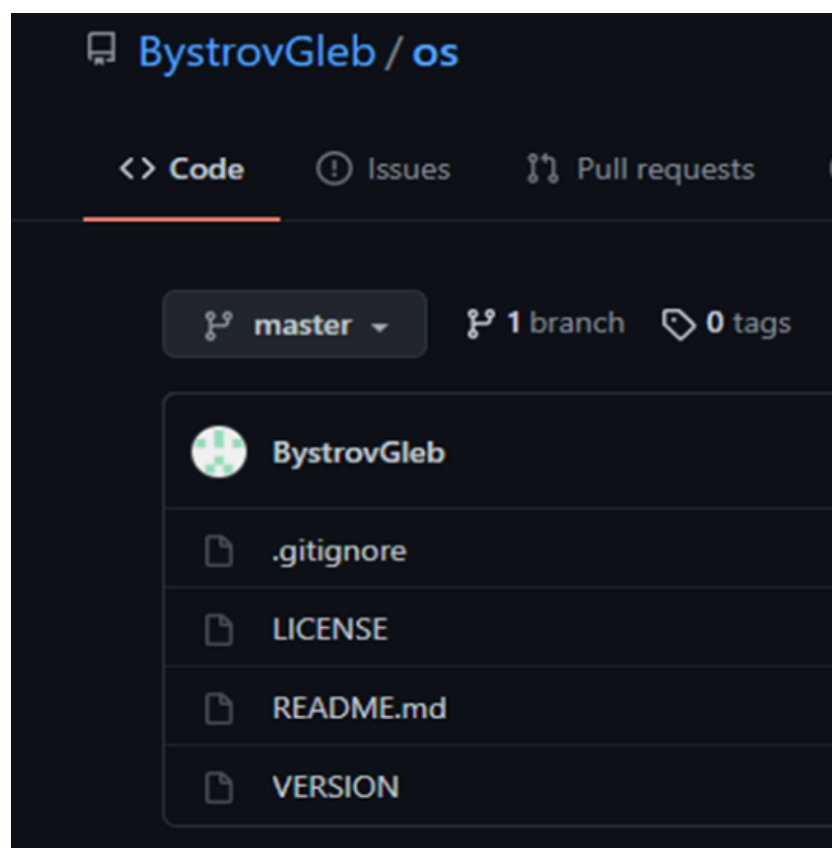


Figure 3.14: Репозиторий

## 4 Контрольные вопросы

1. Что такое системы контроля версий (VCS) и для решения каких задач они предназначаются? Система контроля версий —☒ это система, записывающая изменения в файл или набор файлов в течение времени и позволяющая вернуться позже к определённой версии. Программистам, дизайнерам, разработчикам и другим специалистам будет удобно сохранять разные версии проектов, чтобы легко к ним возвращаться при необходимости. Благодаря системе контроля версий несколько участников могут работать с файлами и смотреть изменения каждого участника.
2. Объясните следующие понятия VCS и их отношения: хранилище, commit, история, рабочая копия.
  - Хранилище – традиционные системы управления версиями используют централизованную модель, когда имеется единое хранилище документов, управляемое специальным сервером, который и выполняет большую часть функций по управлению версиями.
  - commit – завершив очередной этап работы над заданием, разработчик фиксирует свои изменения, передавая их на сервер.
  - история – вся история изменения документов хранится на каждом компьютере, в локальном хранилище, и при необходимости отдельные фрагменты истории локального хранилища синхронизируются с аналогичным хранилищем на другом компьютере.
  - рабочая копия – обычно создаётся локальная копия документа, так называемая «рабочая копия».

3. Что представляют собой и чем отличаются централизованные и децентрализованные VCS? Приведите примеры VCS каждого вида. Централизованные системы контроля версий представляют собой приложения типа клиент-сервер, когда репозиторий проекта существует в единственном экземпляре и хранится на сервере. В децентрализованной системе у каждого есть свой полноценный репозиторий. Децентрализованные системы были созданы для обмена изменениями. При использовании децентрализованной системы нет какой-то жестко заданной структуры репозитория с центральным сервером.
4. Опишите действия с VCS при единоличной работе с хранилищем. Первым действием, которое должен выполнить разработчик, является извлечение рабочей копии проекта или той его части, с которой предстоит работать. Это действие выполняется с помощью команды извлечения версии (обычно checkout или clone). Разработчик задаёт версию, которая должна быть скопирована, по умолчанию обычно копируется последняя (или выбранная администратором в качестве основной) версия. По команде извлечения устанавливается соединение с сервером, и проект (или его часть — один из каталогов с подкаталогами) в виде дерева каталогов и файлов копируется на компьютер разработчика. Работая с проектом, разработчик изменяет только файлы основной рабочей копии. Вторая локальная копия хранится в качестве эталона, позволяя в любой момент без обращения к серверу определить, какие изменения внесены в конкретный файл или проект.
5. Опишите порядок работы с общим хранилищем VCS. Делать мелкие исправления в проекте можно путём непосредственной правки рабочей копии и последующей фиксации изменений прямо в главной ветви (в стволе) на сервере. Для изменений обычной практикой является создание ветвей (branch), то есть «отпочковывание» от ствола в какой-то версии нового варианта проекта или его части, разработка в котором ведётся параллельно с изменениями в основной версии. Ветвь создаётся специальной командой.

Рабочая копия ветви может быть создана заново обычным образом (командой извлечения рабочей копии, с указанием адреса или идентификатора ветви), либо путём переключения имеющейся рабочей копии на заданную ветвь.

6. Каковы основные задачи, решаемые инструментальным средством git? Система спроектирована как набор программ, специально разработанных с учётом их использования в сценариях. Это позволяет удобно создавать специализированные системы контроля версий на базе Git или пользовательские интерфейсы. Git поддерживает быстрое разделение и слияние версий, включает инструменты для визуализации и навигации по нелинейной истории разработки. Предоставляет каждому разработчику локальную копию всей истории разработки, изменения копируются из одного репозитория в другой. Удалённый доступ к репозиториям Git обеспечивается сервером.
7. Назовите и дайте краткую характеристику командам git. Наиболее часто используемые команды git: – создание основного дерева репозитория: `git init` – получение обновлений (изменений) текущего дерева из центрального репозитория: `git pull` – отправка всех произведённых изменений локального дерева в центральный репозиторий: `git push` – просмотр списка изменённых файлов в текущей директории: `git status` – просмотр текущих изменений: `git diff` – добавить все изменённые и/или созданные файлы и/или каталоги: `git add`.
8. Приведите примеры использования при работе с локальным и удалённым репозиториями. Локальный репозиторий — она же директория “.git”. В ней хранятся коммиты и другие объекты. Удаленный репозиторий – репозиторий который считается общим, в который вы можете передать свои коммиты из локального репозитория, что бы остальные могли их увидеть.
9. Что такое и зачем могут быть нужны ветви (branches)? Ветки нужны для того, чтобы участники могли вести совместную работу над проектом и не

мешать друг другу при этом. При создании проекта, Git создает базовую ветку. Она называется master веткой. Она считается центральной веткой, т.е. в ней содержится основной код приложения.

10. Как и зачем можно игнорировать некоторые файлы при commit? Часто в рабочей директории появляются файлы, которые нужно игнорировать. Зачастую, у имеется группа файлов, которые вы не только не хотите автоматически добавлять в репозиторий, но и видеть в списках не отслеживаемых. К таким файлам обычно относятся автоматически генерируемые файлы. В таком случае, вы можете создать файл .gitignore с перечислением шаблонов соответствующих таким файлам. Это защитит вас от случайного добавления в репозиторий файлов, которых вы там видеть не хотите.

## **5 Выводы**

В данной лабораторной работе мне успешно удалось приобрести практические навыки применения средств контроля версий. Также я смог научиться работать с системой контроля версий Git с помощью командной строки.