**Міністерство освіти і науки України**

**Харківський національний університет ім. В. Н. Каразіна**

**Факультет комп’ютерних наук**

**Кафедра безпеки iнформацiйних систем i технологiй**

**Лабораторна робота № 1**

*з навчальної дисципліни*

«**Математичні методи та технології тестування і верифікації програмного забезпечення**»

***Виконала:***

студентка групи КБ-22

Рогоза П. В.

***Перевірила:***

Мелкозьорова О. М.

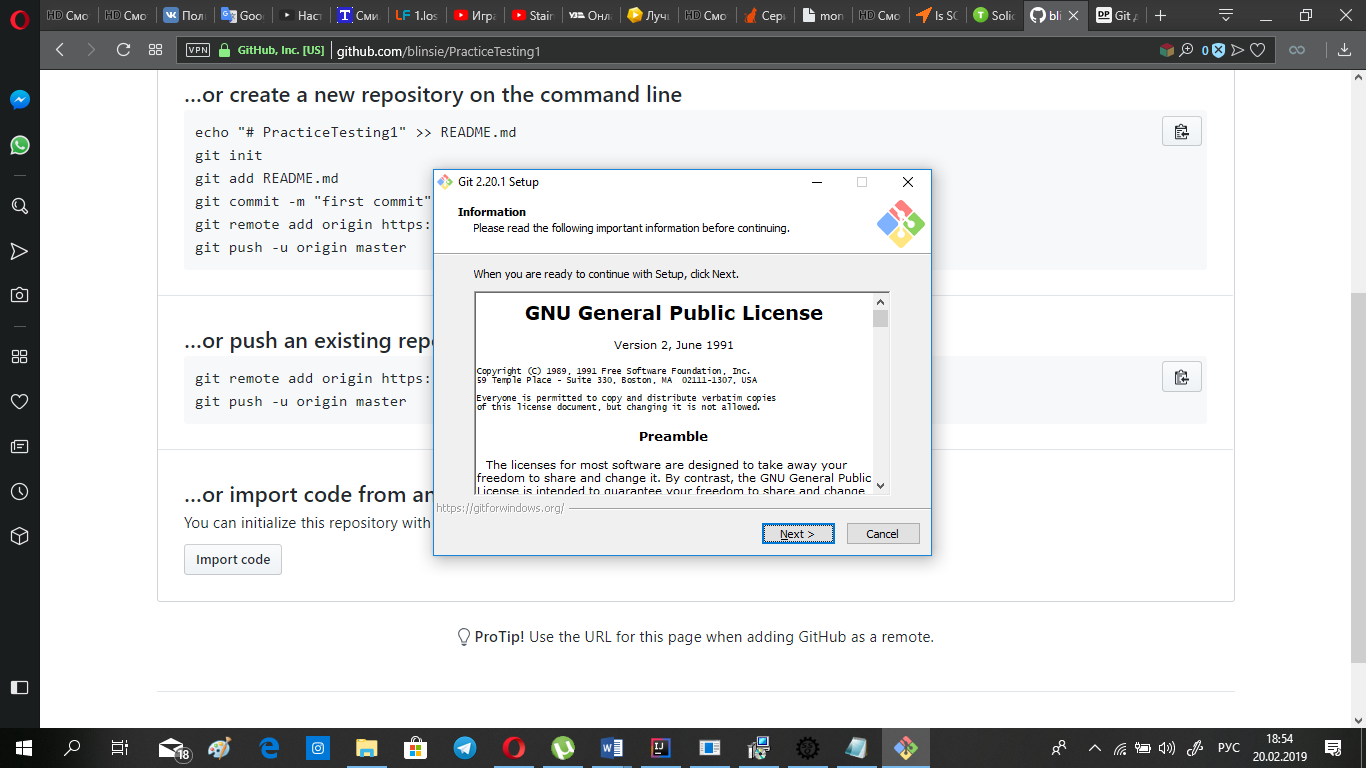
Харків – 2019 р.

**ЛАБОРАТОРНАЯ РАБОТА № 1**

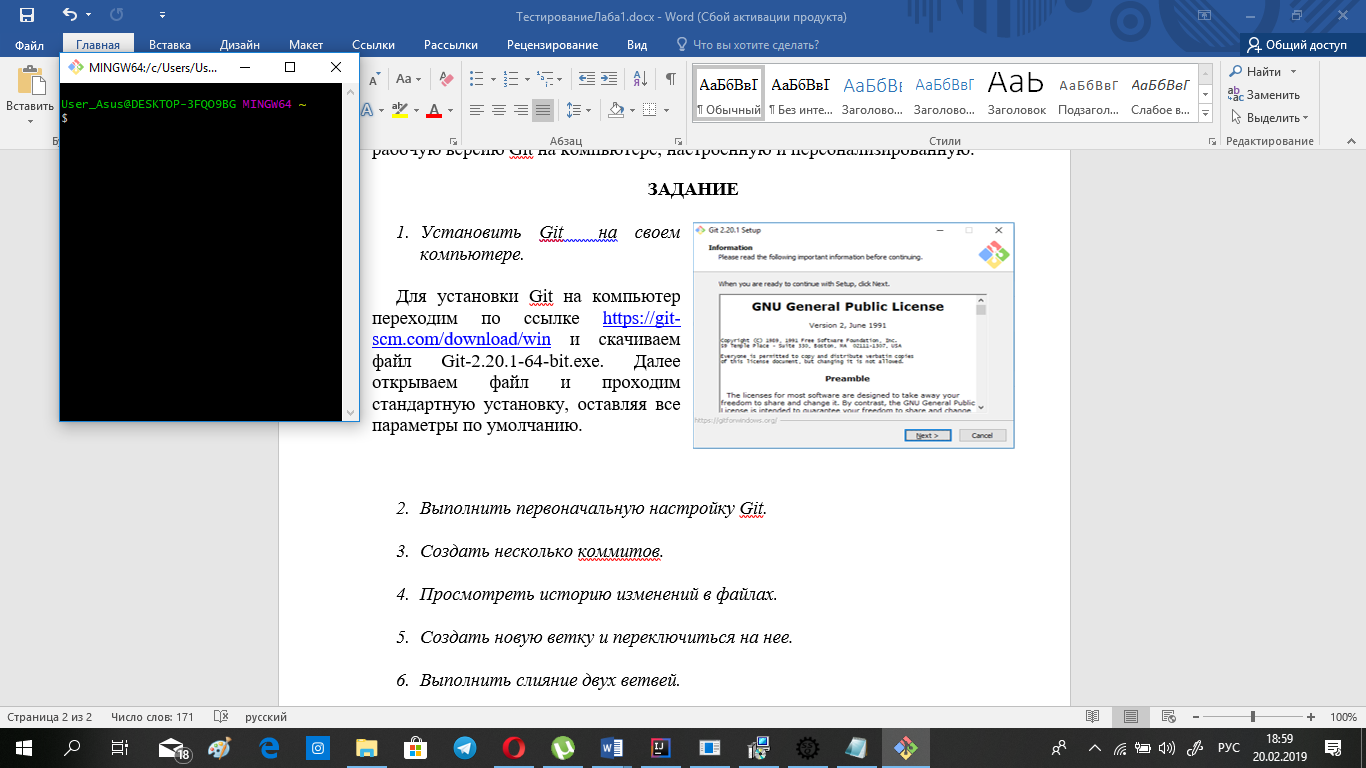
**Тема:** Изучение системы контроля версии на примере Git.

**Цель работы:** Цель – получить базовые знания о том, что такое Git и чем он отличается от централизованных систем контроля версий. Также получить рабочую версию Git на компьютере, настроенную и персонализированную.

**ЗАДАНИЕ**

1. *Установить Git на своем компьютере.*

Для установки Git на компьютер переходим по ссылке <https://git-scm.com/download/win> и скачиваем файл Git-2.20.1-64-bit.exe. Далее открываем файл и проходим стандартную установку, оставляя все параметры по умолчанию.

Ждем, пока установка закончится, запускаем приложение и видим консоль, готовую к работе:

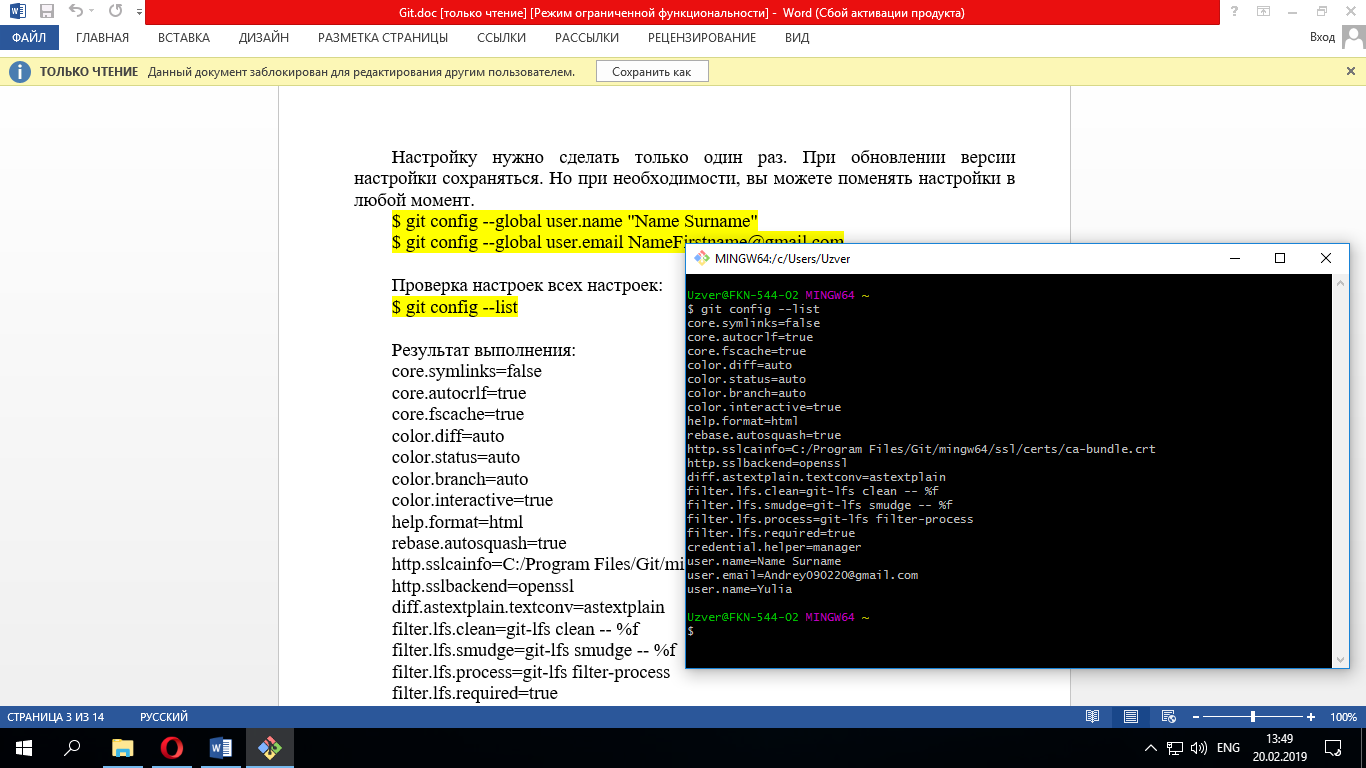
1. *Выполнить первоначальную настройку Git.*

Для настройки GIT требуется прописать команды:

**$ git config --global user.name "Name Surname"**

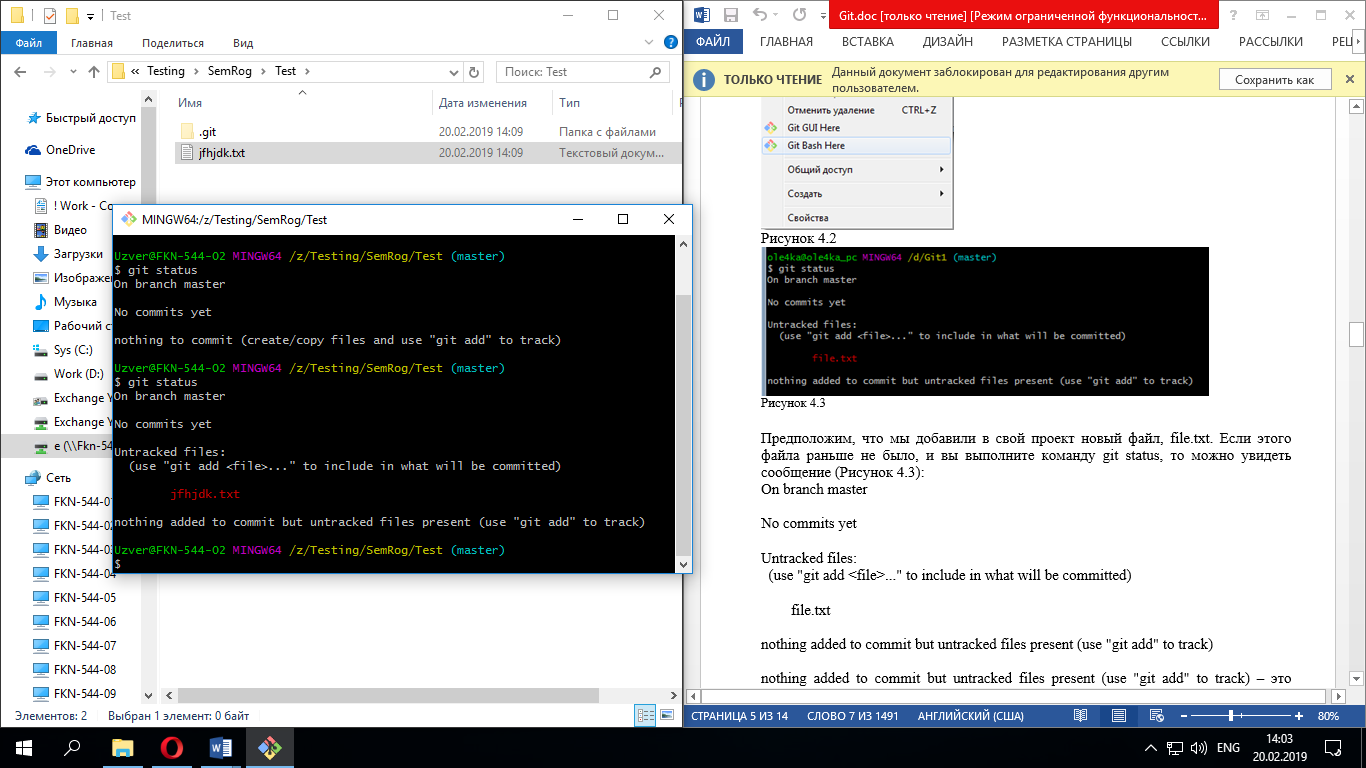
**$ git config --global user.email** [**NameFirstname@gmail.com**](mailto:NameFirstname@gmail.com) – указав свои данные.

Для проверки всех настроек выполнить команду: **$ git config –list**



1. *Создать несколько коммитов.*

Перед созданием коммита можно проверить, были ли перед этим совершены другие коммиты, пользуясь командой: **$ git status.**

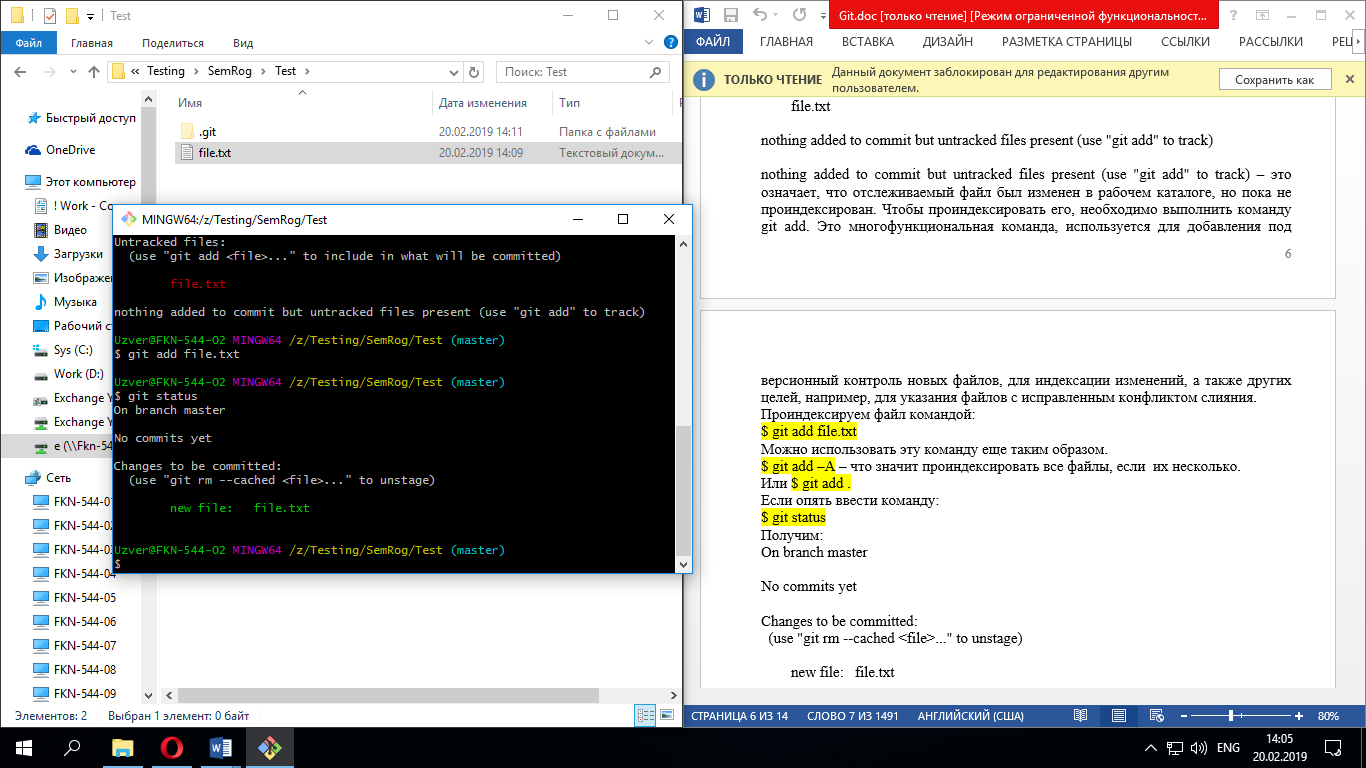


Для осуществления коммита, требуется создать файл (или несколько) и добавить их в корневую папку. После создания коммита данные обновятся (команда: **$ git status**), и программа покажет созданные коммиты.

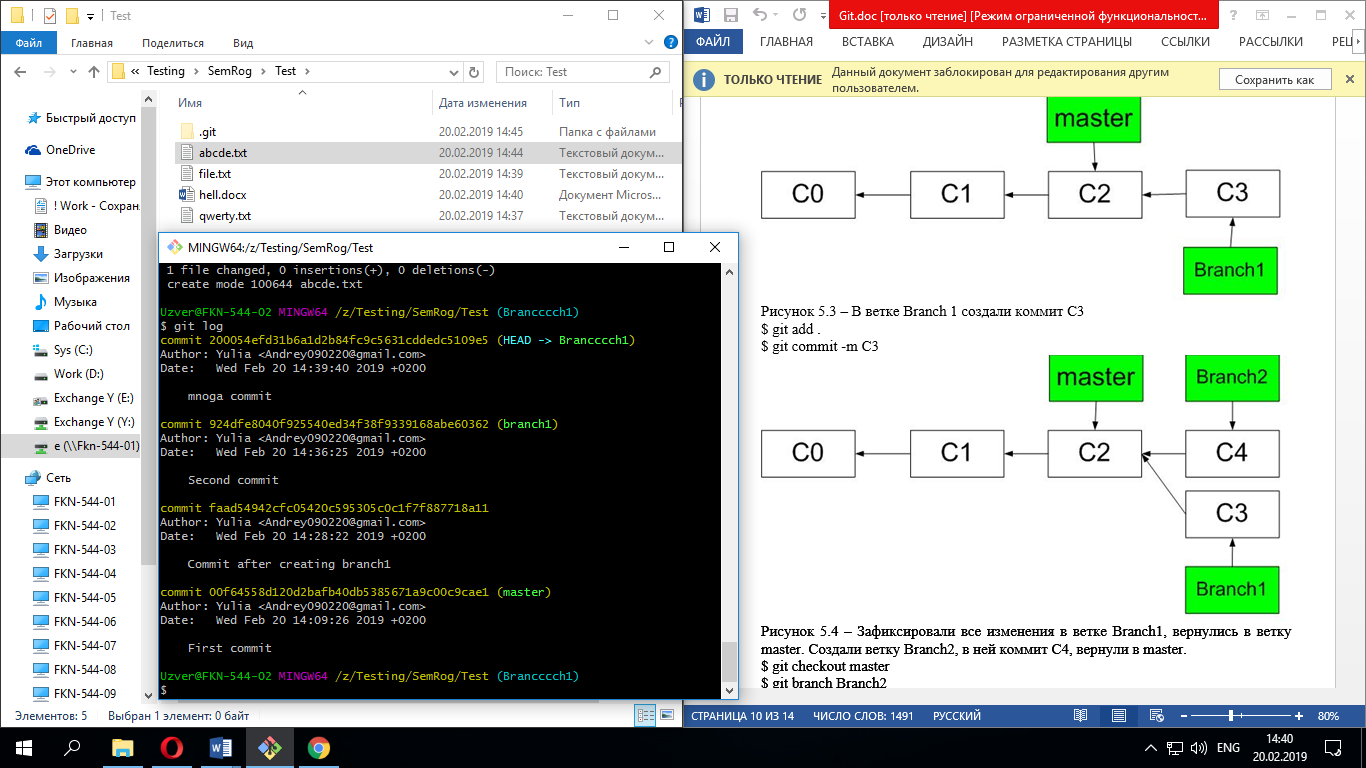
1. *Просмотреть историю изменений в файлах.*

Для добавления файла под версионный контроль нужно вызвать команду:

**$ git add file.txt** (file – имя файла).

****

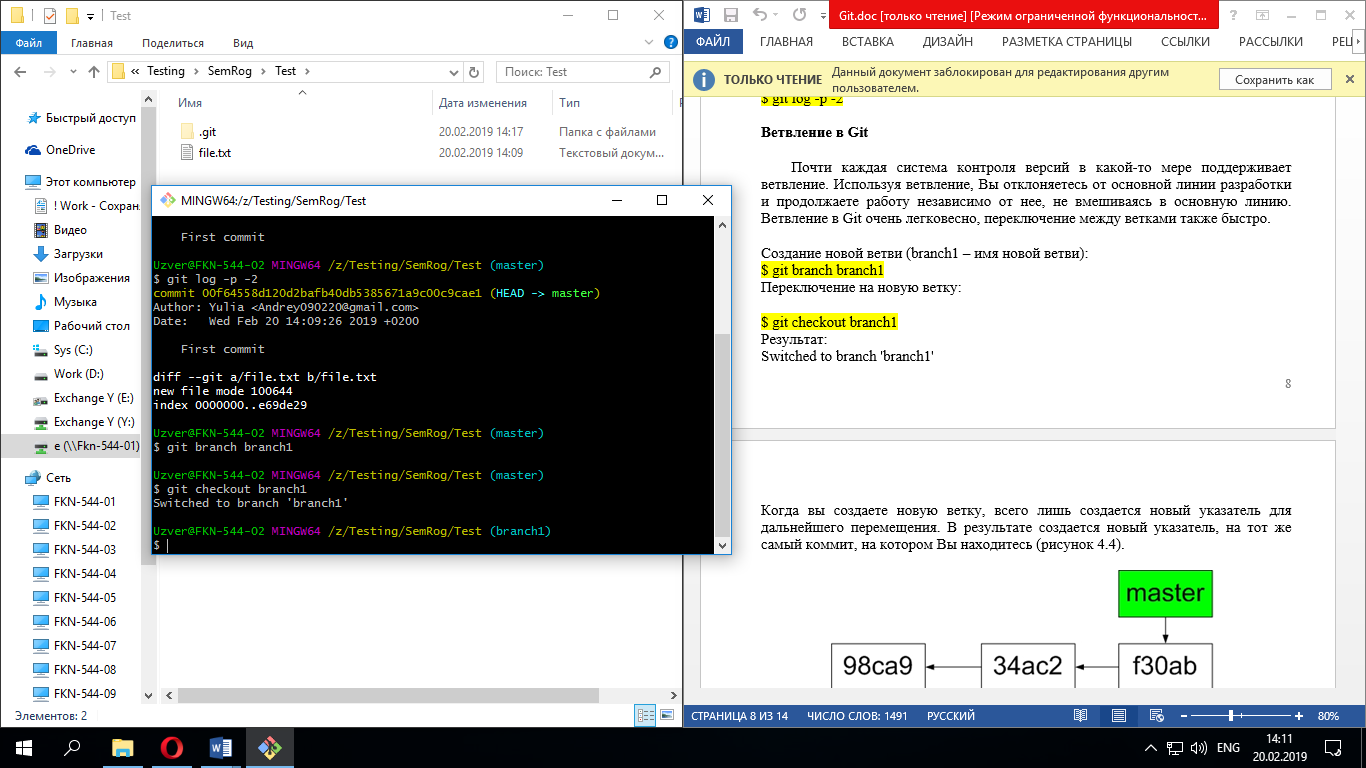
Для просмотра истории используется команда: **$ git log**



1. *Создать новую ветку и переключиться на нее.*

Новая ветка создается командой: **$ git branch Название\_ветки**

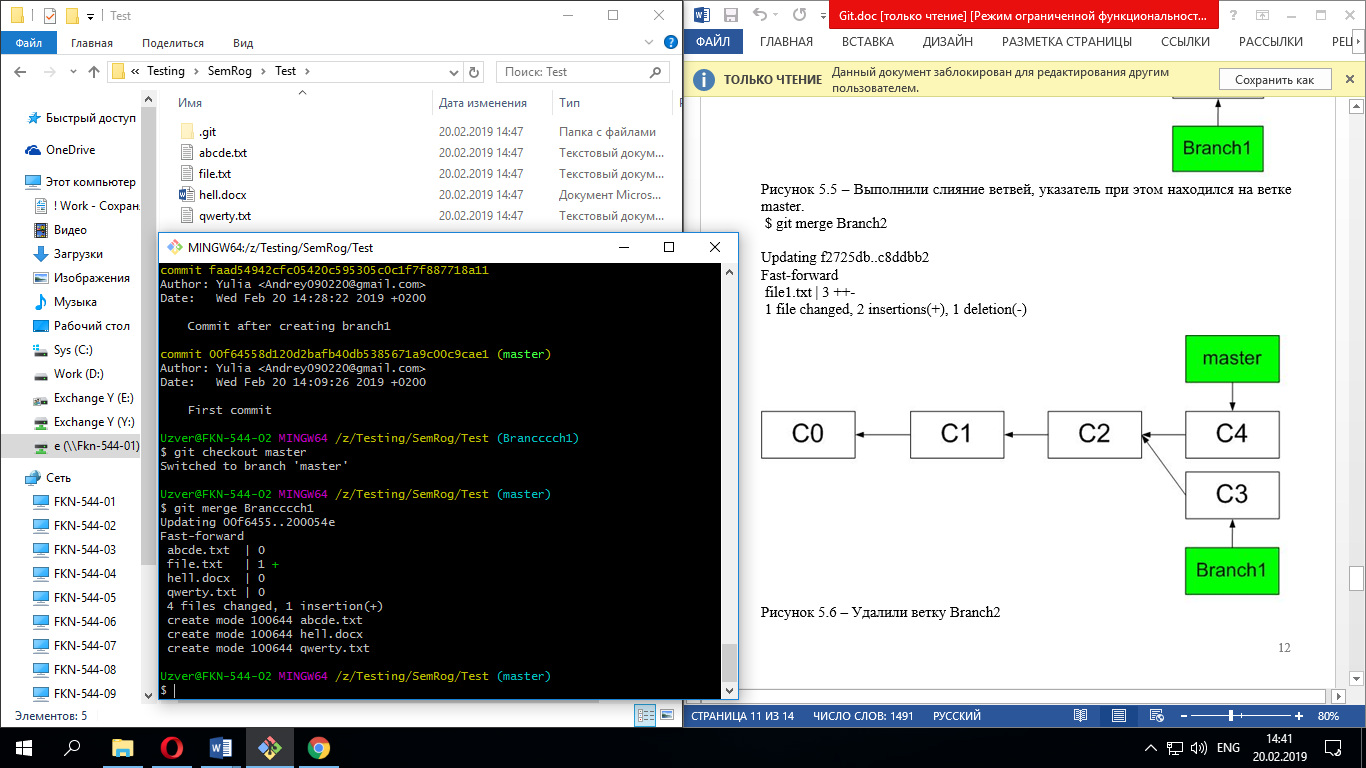
Команда переключения на другую ветку: **$ git checkout Название\_ветки**

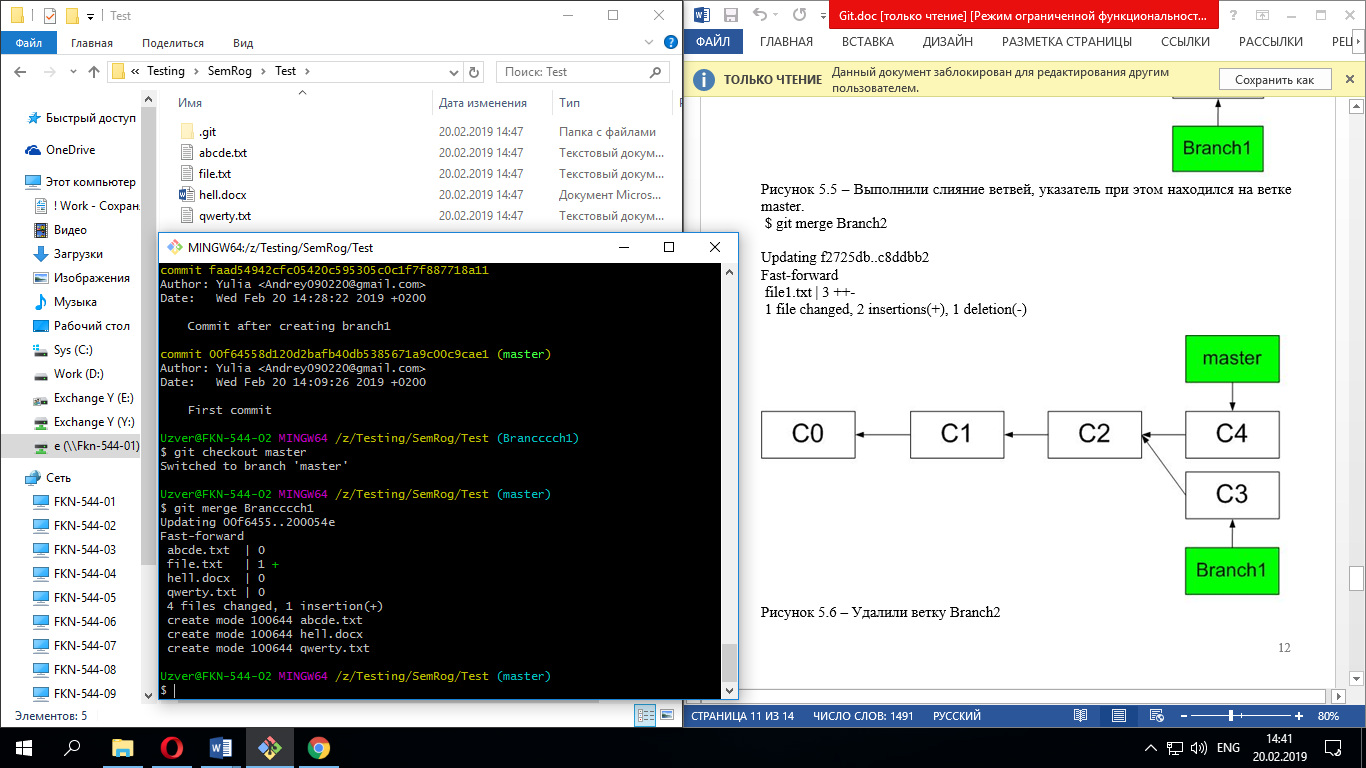


1. *Выполнить слияние двух ветвей.*

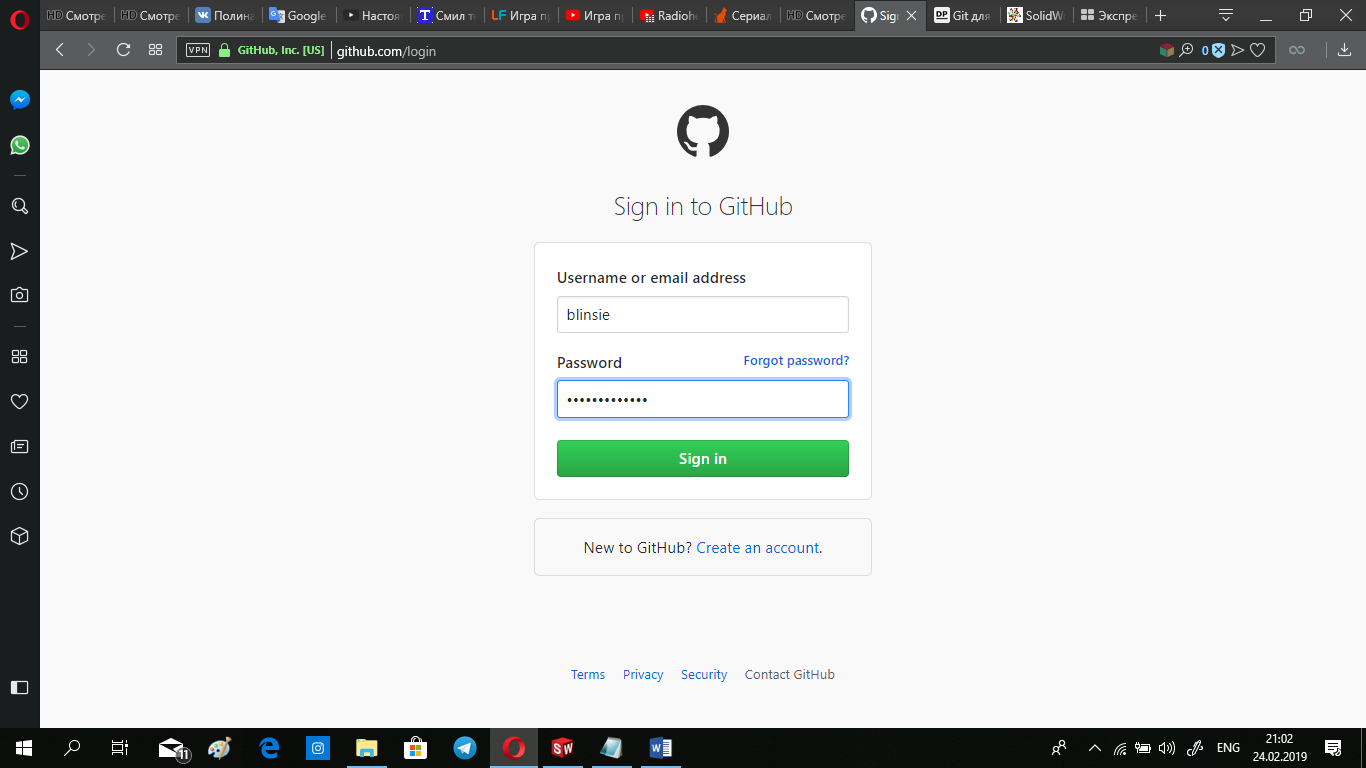
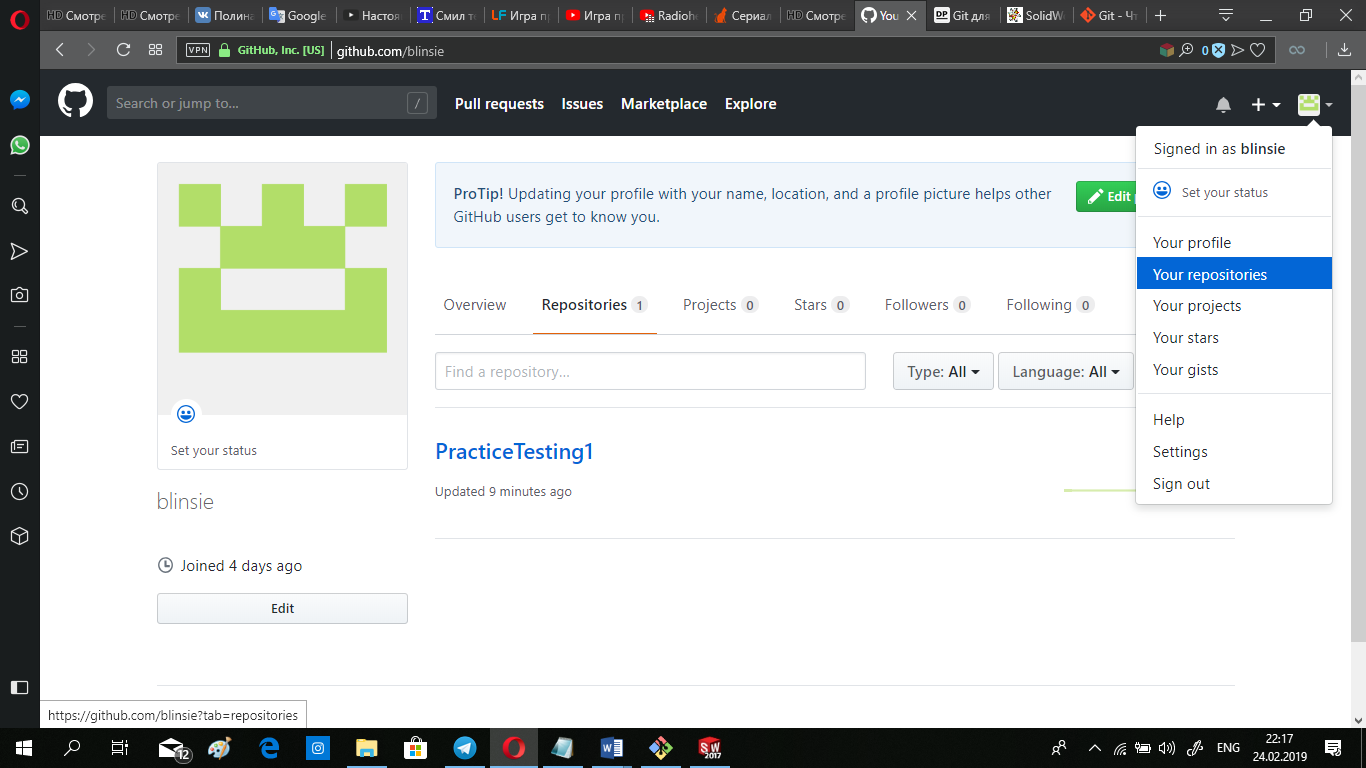
Слияние ветвей осуществляется с помощью команды:

**$ git merge Ветка\_с\_которой\_соединяется\_текущая**

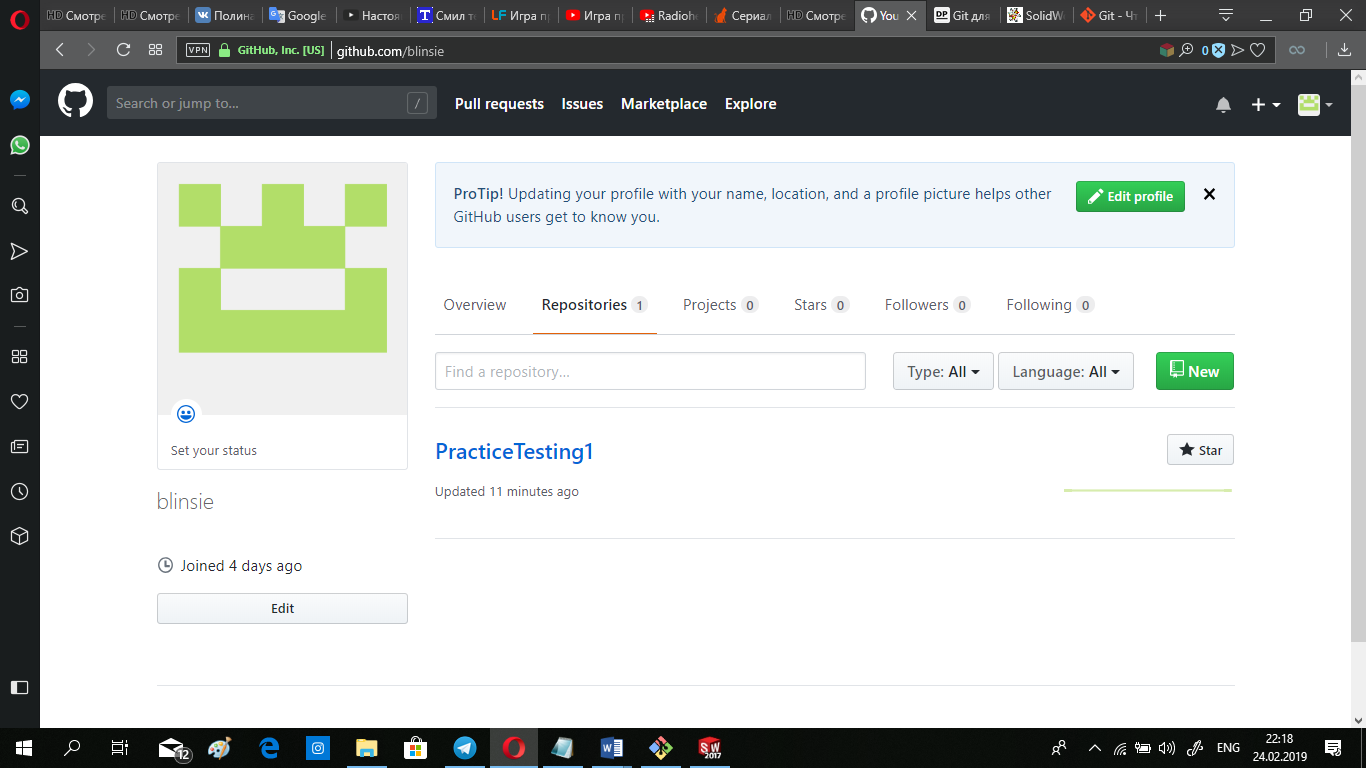




1. *Зарегистрироваться на:* [*https://github.com/*](https://github.com/)*.*

Для регистрации требуется перейти по ссылке на сайт GitHub, в полях для регистрации указать Имя Пользователя, Почту и Пароль. Подтверждаем регистрацию, проверяем почту.

Далее заходим в личный кабинет, и с помощью функции *New* в разделе *Your Repositories* создаем новый репозиторий *PracticeTesting1* для нашего проекта.

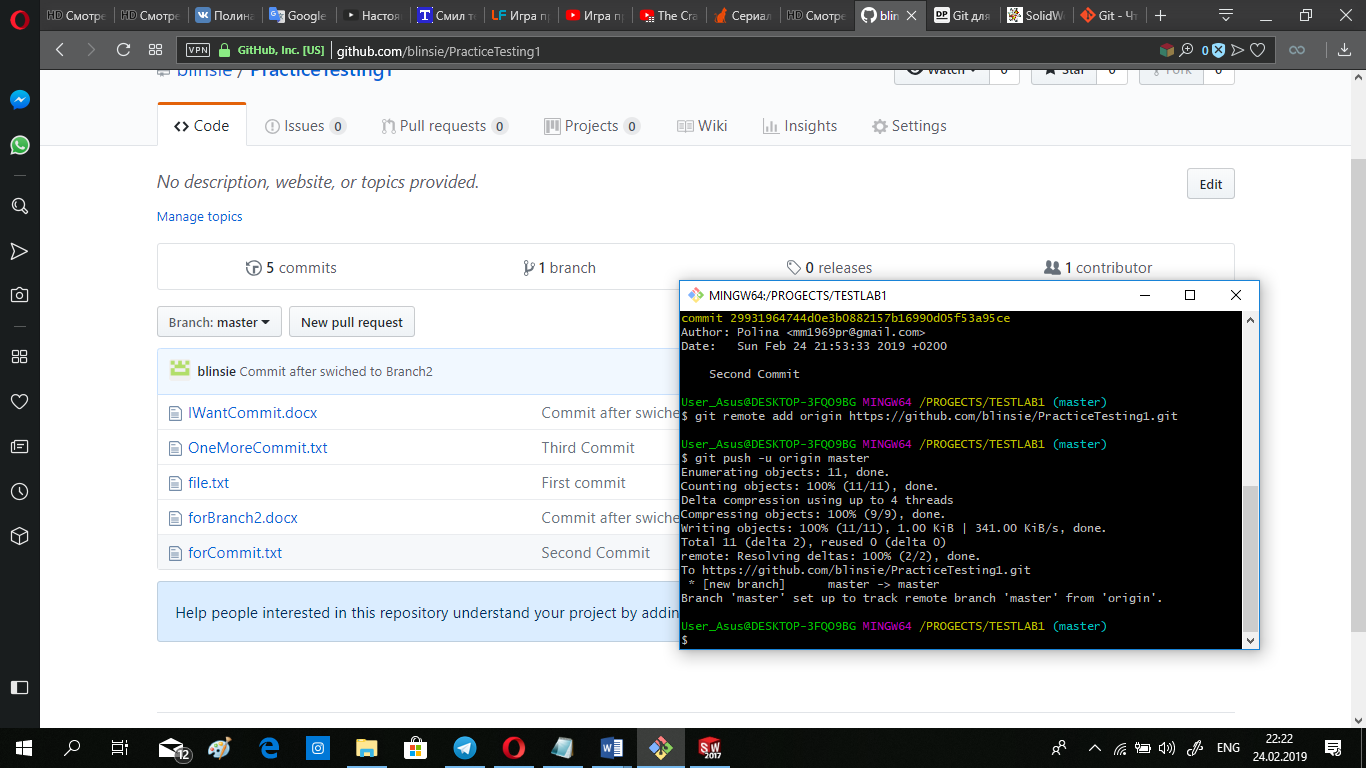


1. *Создать проект и залить все на гит на сервере.*

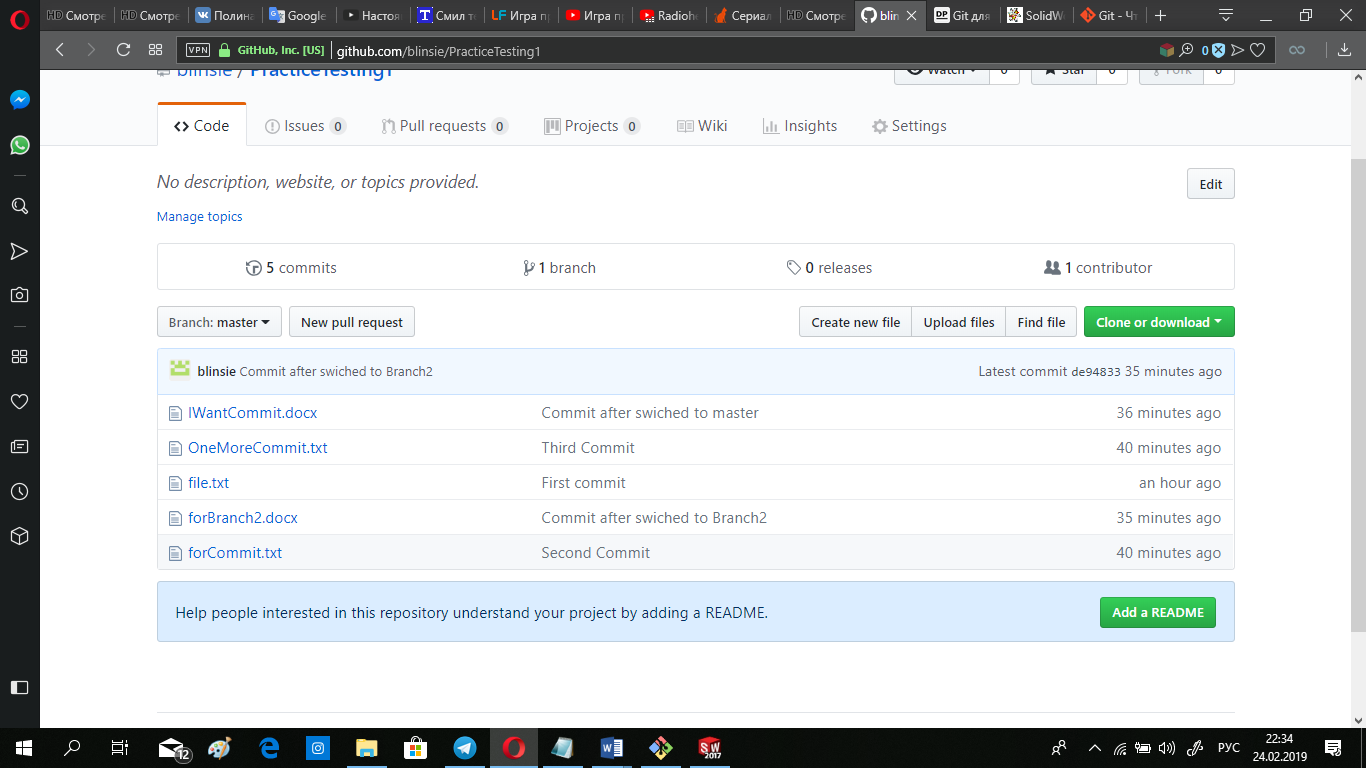
Для того, чтобы залить проект на сайт GitHub, достаточно воспользоваться двумя командами:

**$ git remote add origin** [**https://github.com/blinsie/PracticeTesting1.git**](https://github.com/blinsie/PracticeTesting1.git)

**$ git push -u origin master**



После ввода команд появляется всплывающее окно, в котором нужно ввести данные пользователя (Имя и Пароль). Затем программа показывает успешный результат выполнения запроса, и в репозитории можно увидеть все созданные файлы и коммиты.



*Примечание:* Все файлы и коммиты после работы в аудитории создавались заново на новом компьютере, и имеют другие названия.

**ВЫВОД**

Выполнив лабораторную работу, мы изучили базовые возможности и команды Git’a. Установили рабочую программу, настроили и персонализировали ее, а также создали первый проект и залили его на сайт GitHub.com.

В ходе выполнения работы мы рассмотрели несколько новых понятий:

*Система контроля версий (СКВ)* — это система, регистрирующая изменения в одном или нескольких файлах с тем, чтобы в дальнейшем была возможность вернуться к определённым старым версиям этих файлов. *Централизированная СКВ* же имеет центральный сервер, на котором хранятся все файлы под версионным контролем, и ряд клиентов, которые получают копии файлов из него. К таким системам относятся: CVS, Subversion и Perforce.

В таких системах как Git *(Распределенные СКВ*) клиенты не просто выгружают последние версии файлов, а полностью копируют весь репозиторий, что помогает избежать серверных проблем, копируя каждый сеанс работы, и в случае чего восстановить все копии проекта.