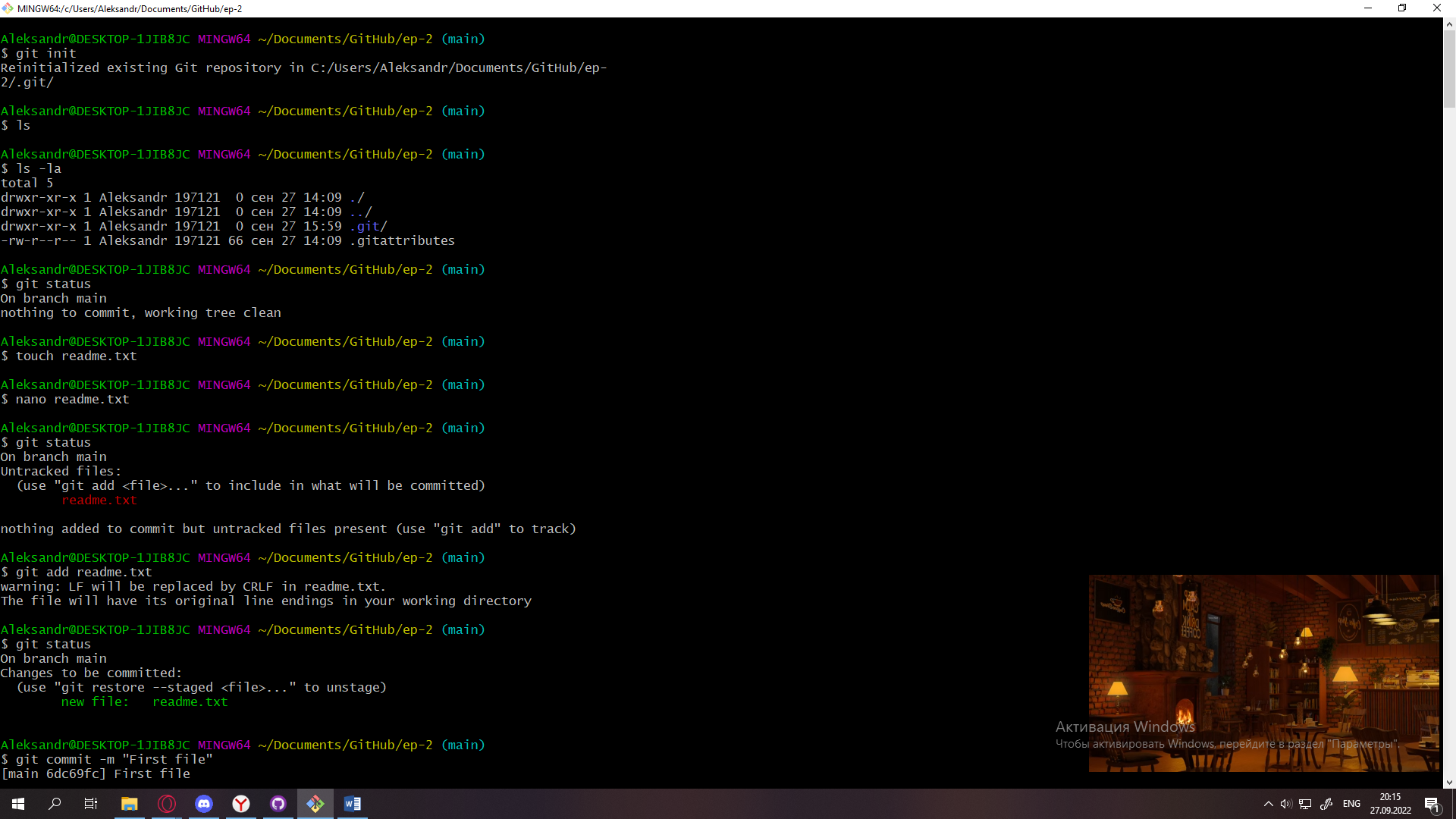
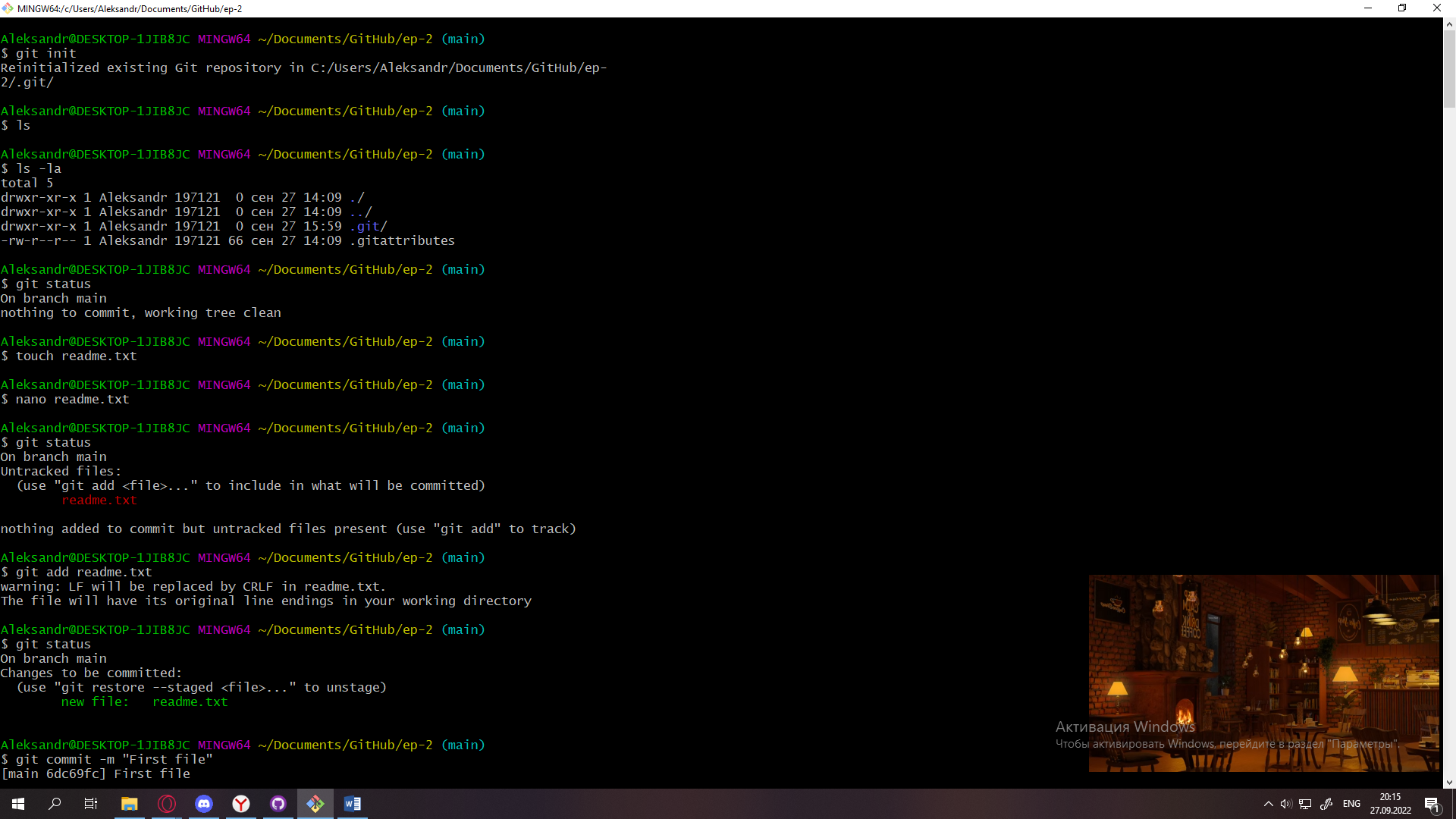
Основные операции, которые могут быть выполнены с системой контроля версий Git. Обзор базовых команд.

При работе я использовала окно терминала Git Bash, GitHub Desktop, папка с подготовленным репозиторием, справочный материал.

Для инициализации системы контроля версий Git используется команда git init.

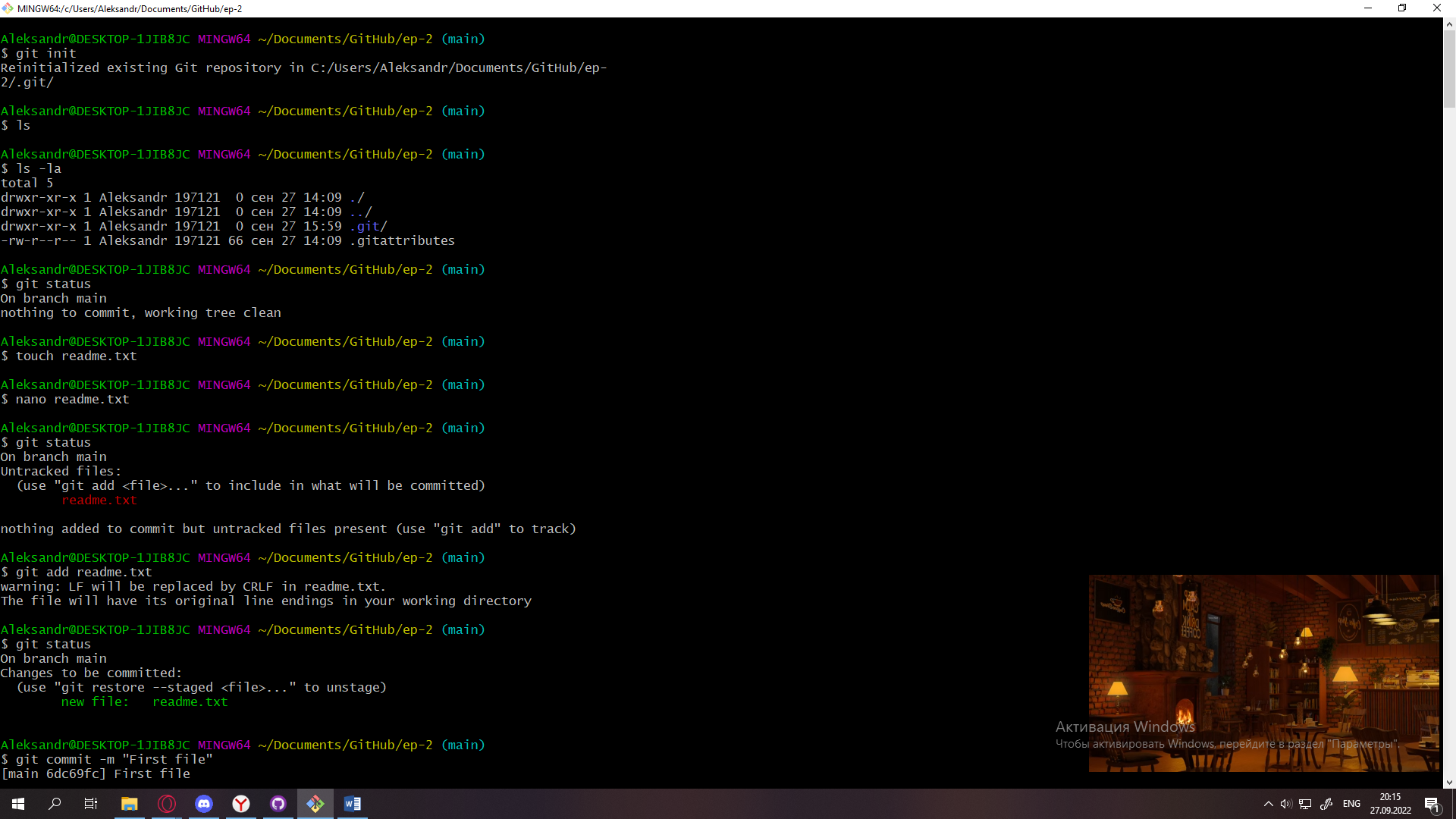


C помощью команд ls, ls –la мы можем увидеть скрытую инициализированную только что папку.



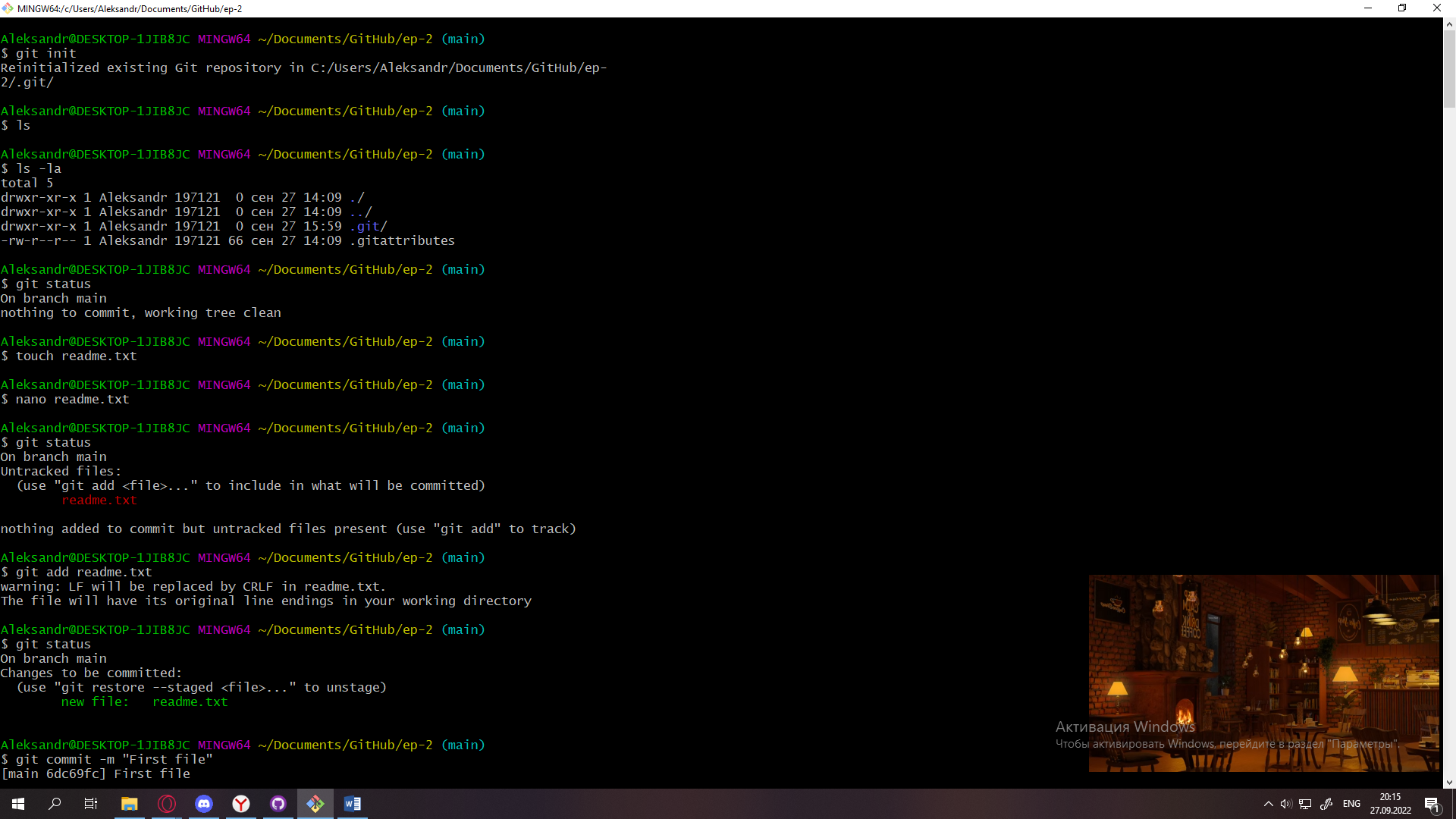
Внутри этой скрытой папки будут храниться элементы системы контроля версий Git.

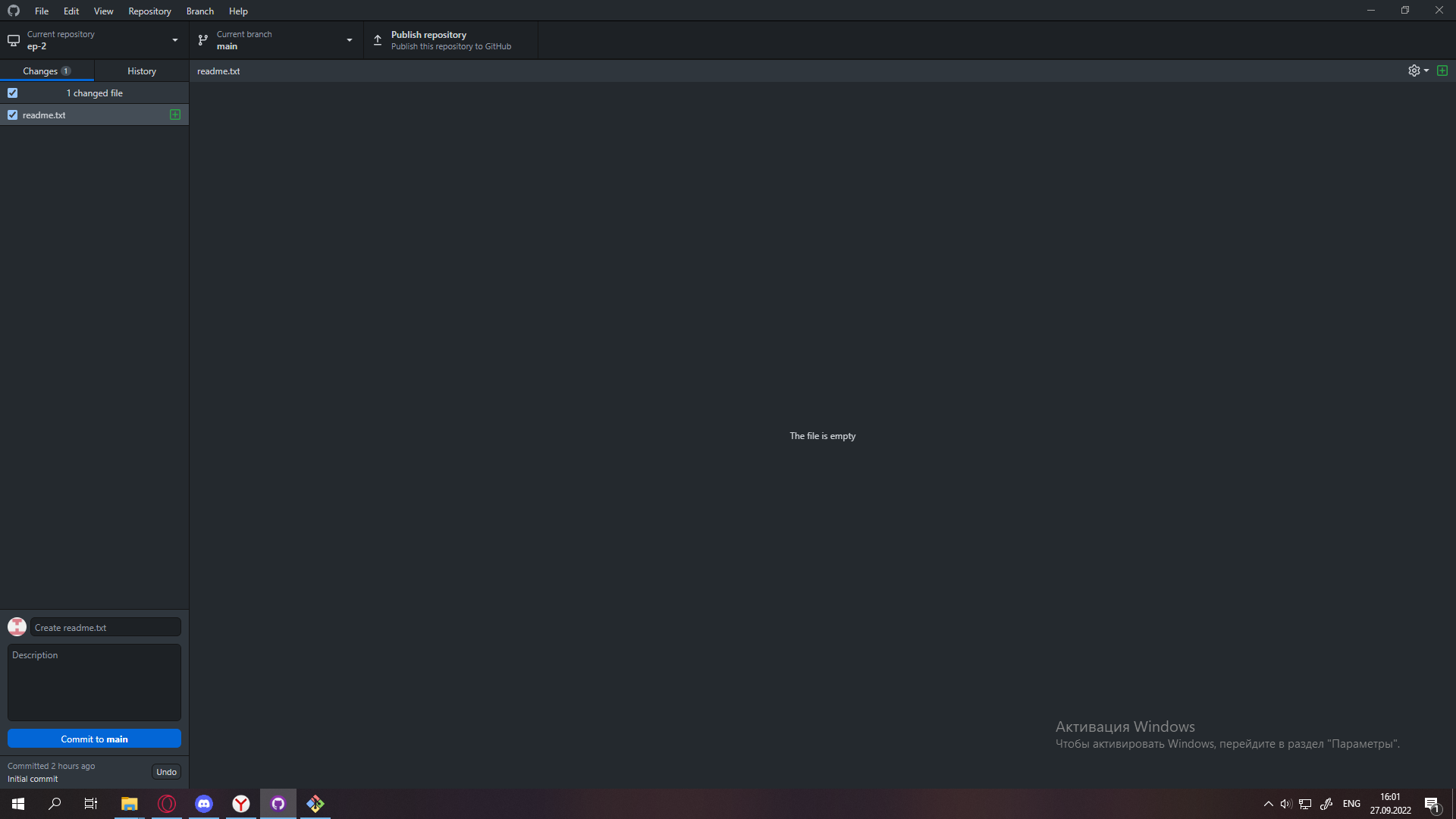
Чтобы определить статут репозитория необходимо ввести команду git status.



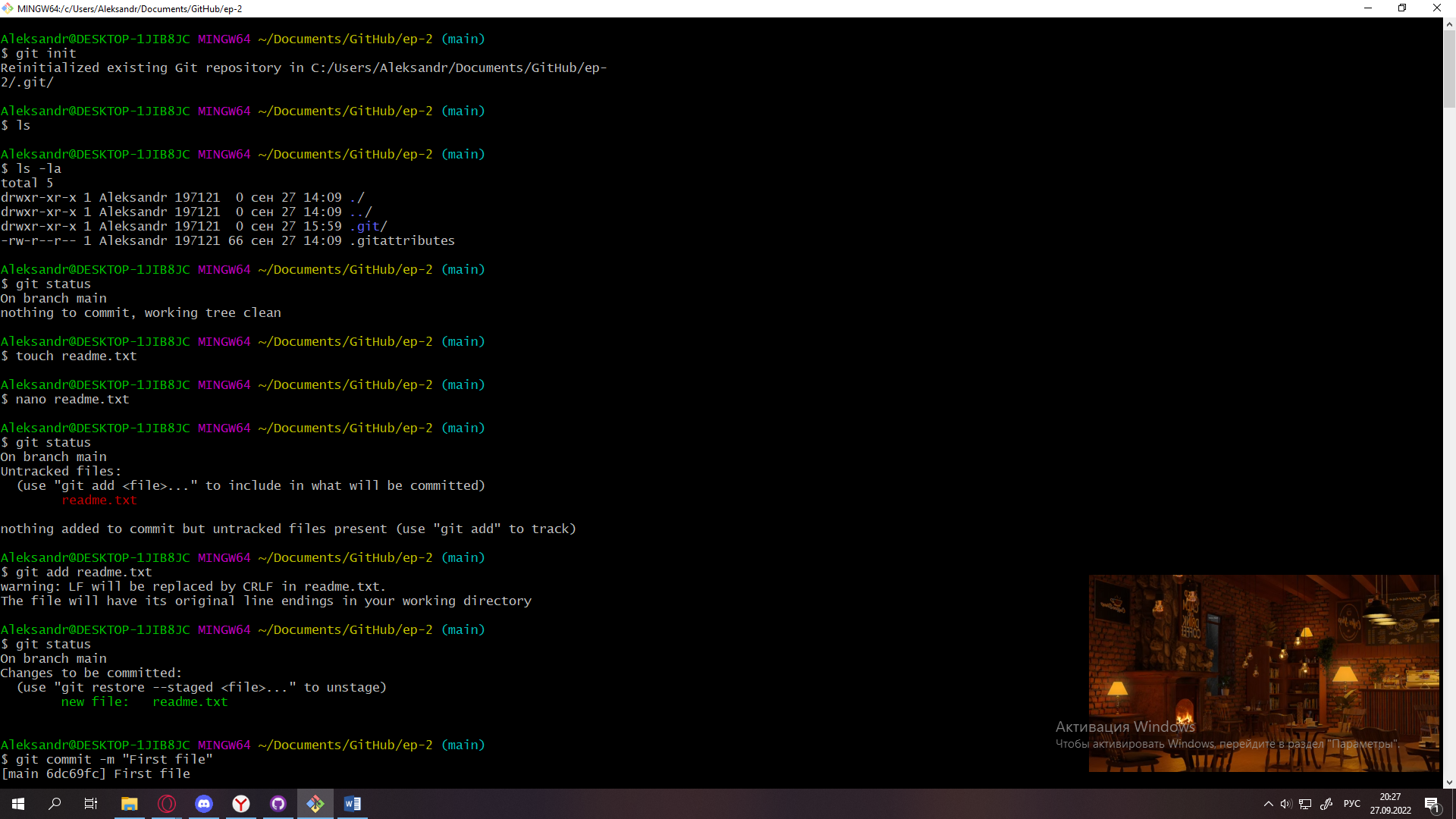
Так как я еще ни с чем не работала, данный каталог пустой, ничего подтверждать не надо, никаких действий совершать не надо.

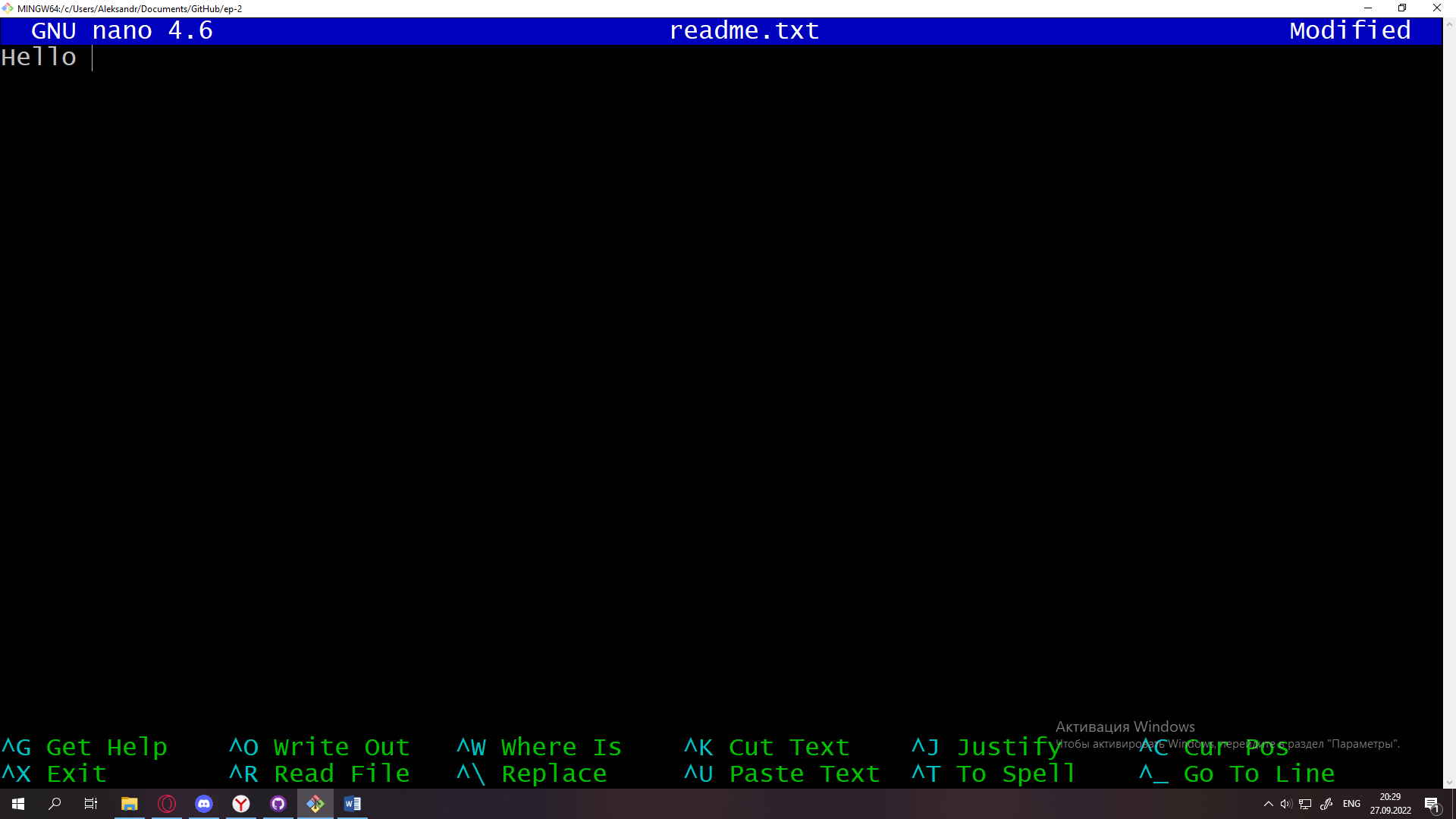
Создадим файл readme.txt с помощью команды touch.



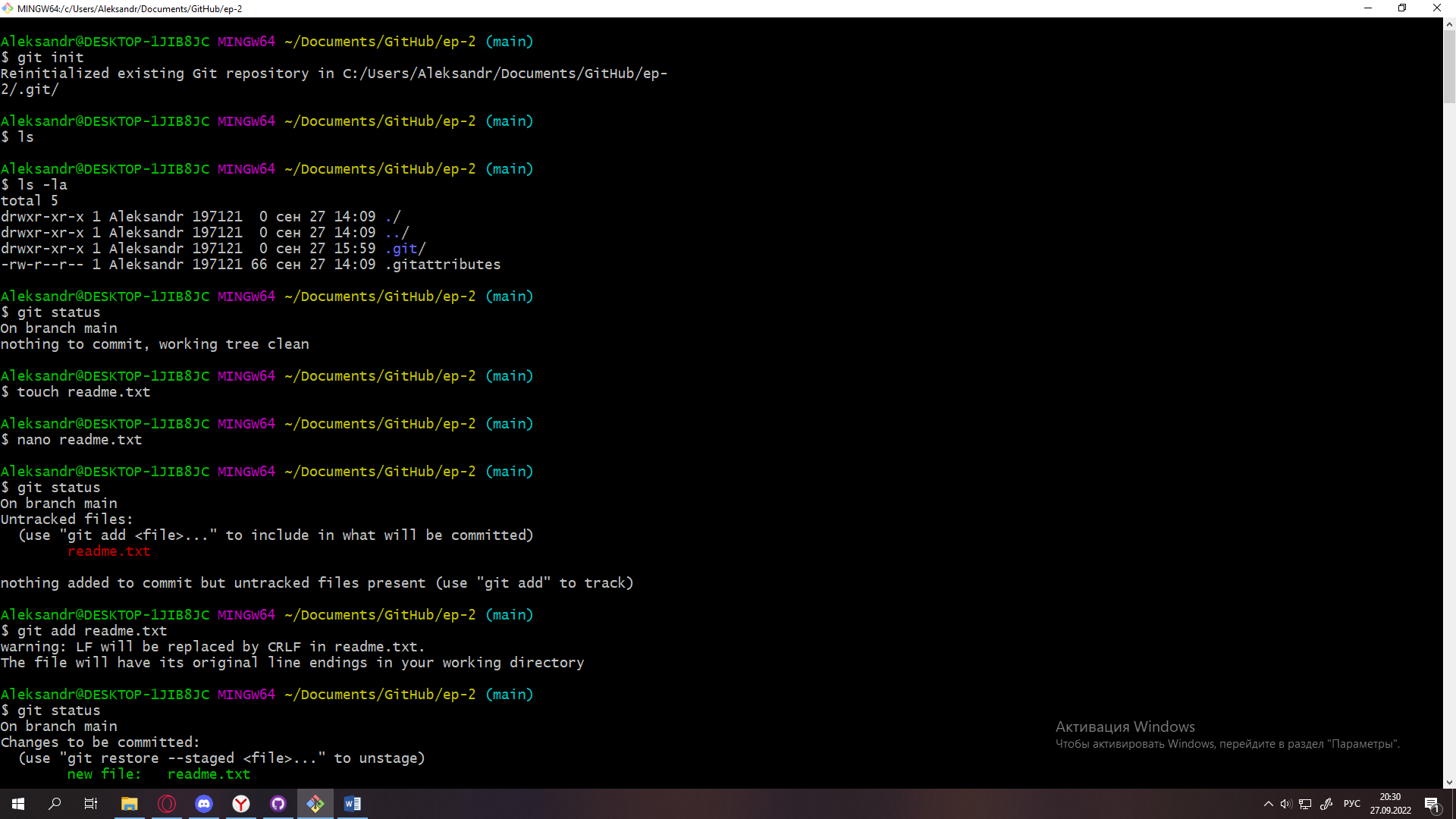


С помощью команды nano можно ввести какой-то текст в существующий файл.

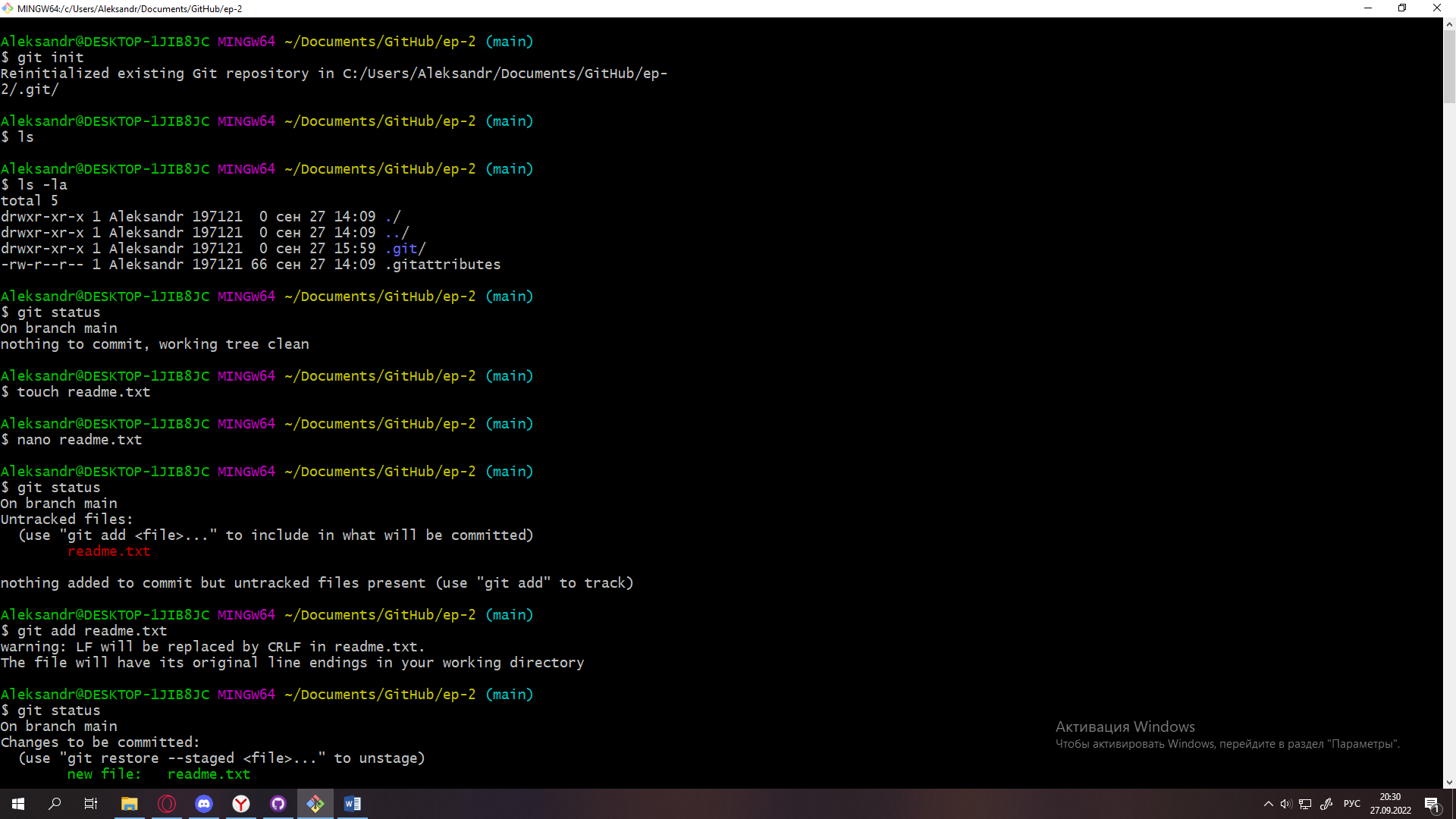




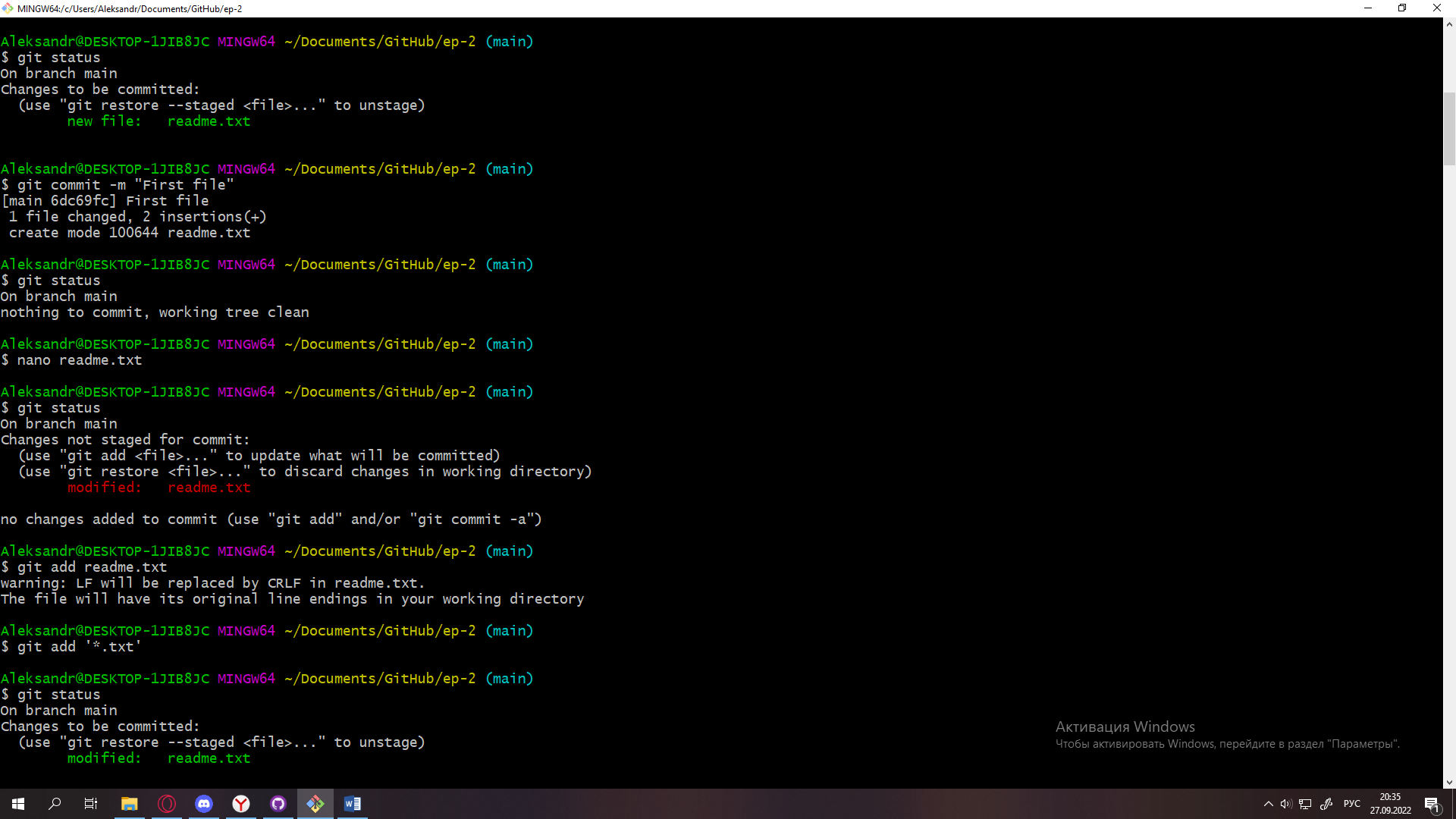
Проверим статус:



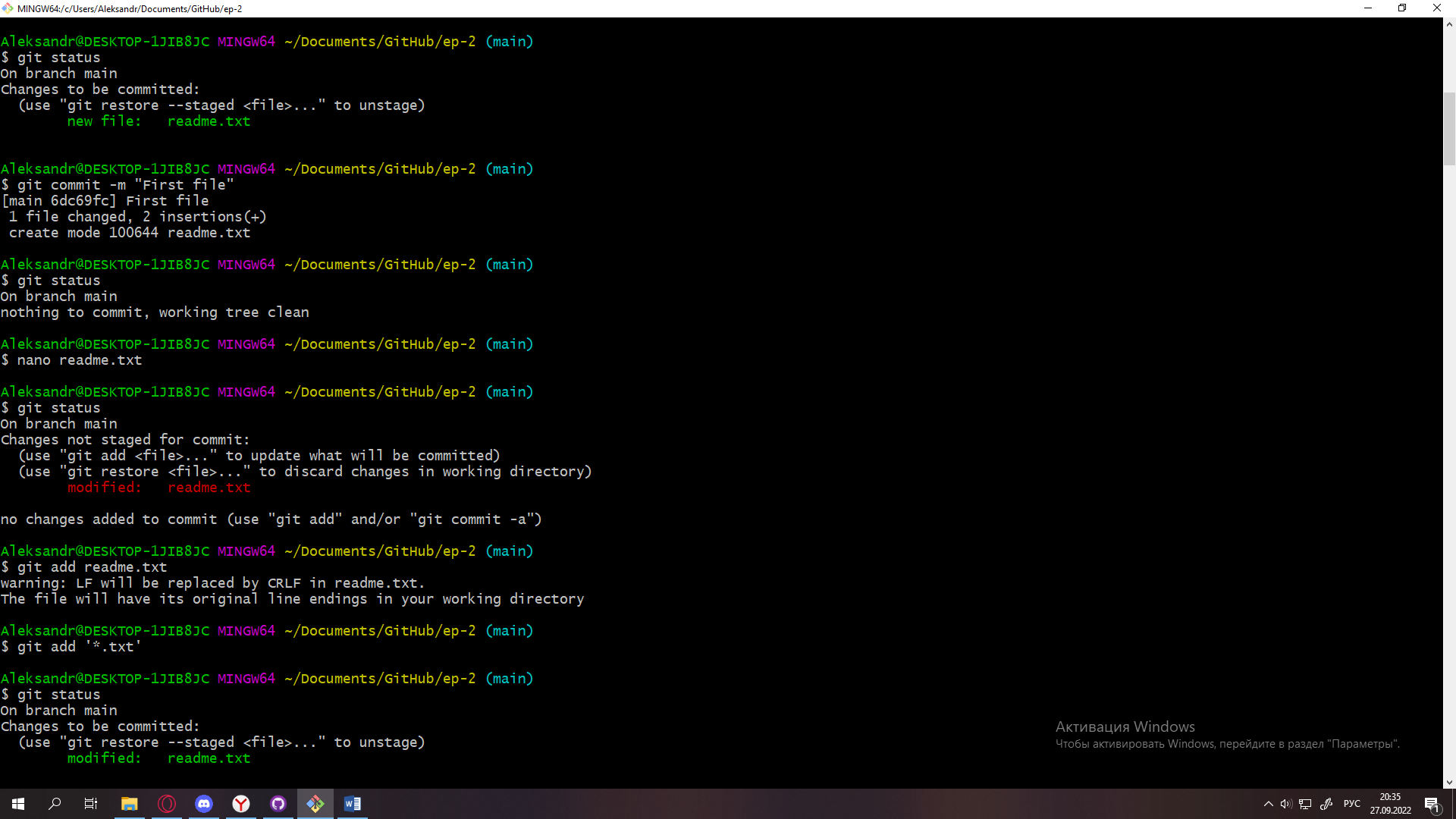
Можно увидеть, что файл был создан, однако он не отслеживается. Отслеживать файл можно с помощью команды git add



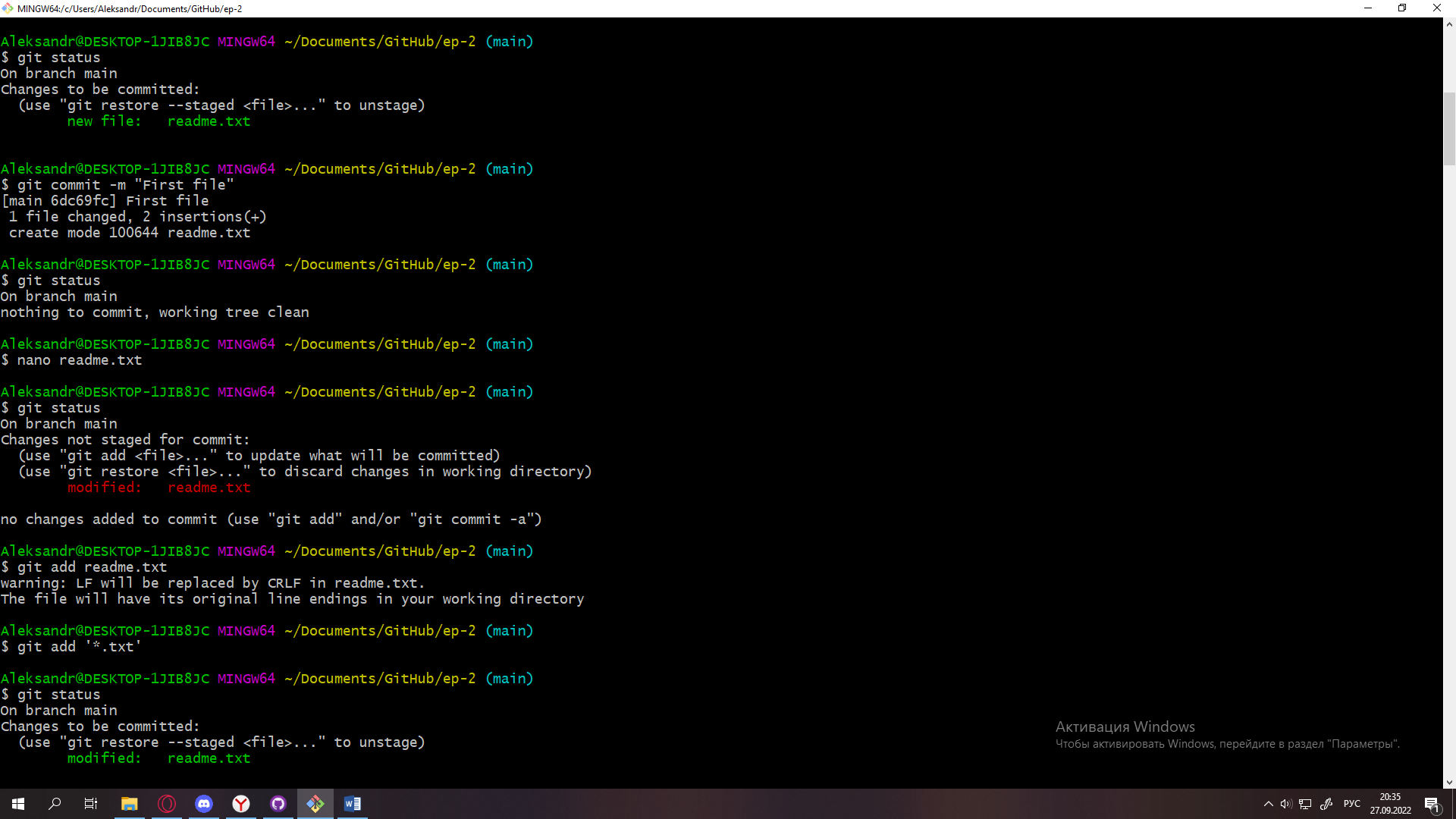
Теперь файл является отслеживаемым.



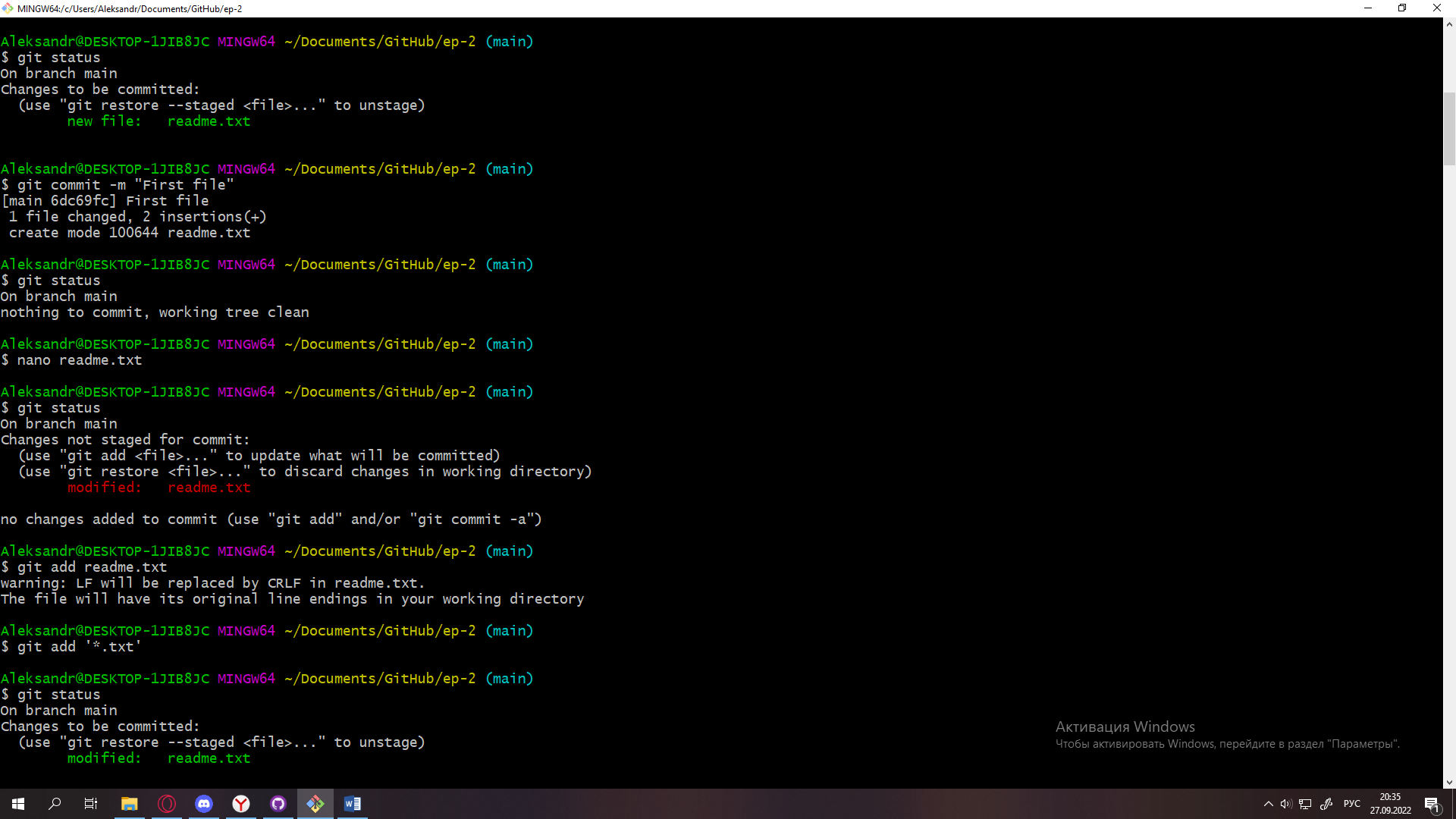
Зафиксируем данный файл с помощью команды git commit –m “[комментарий]”

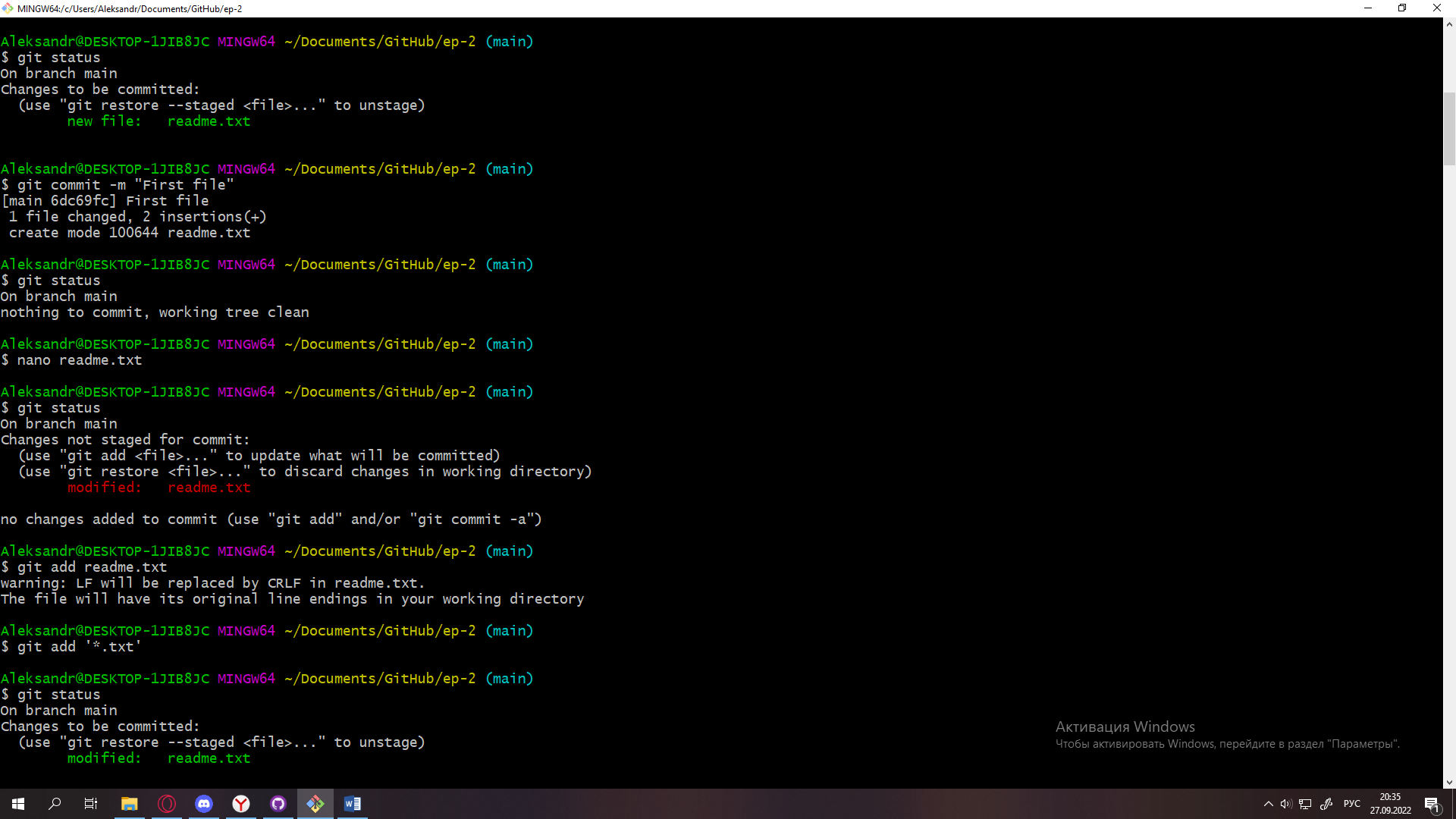


Теперь пока не добавлено новых файлов, нам нечего отслеживать или фиксировать.

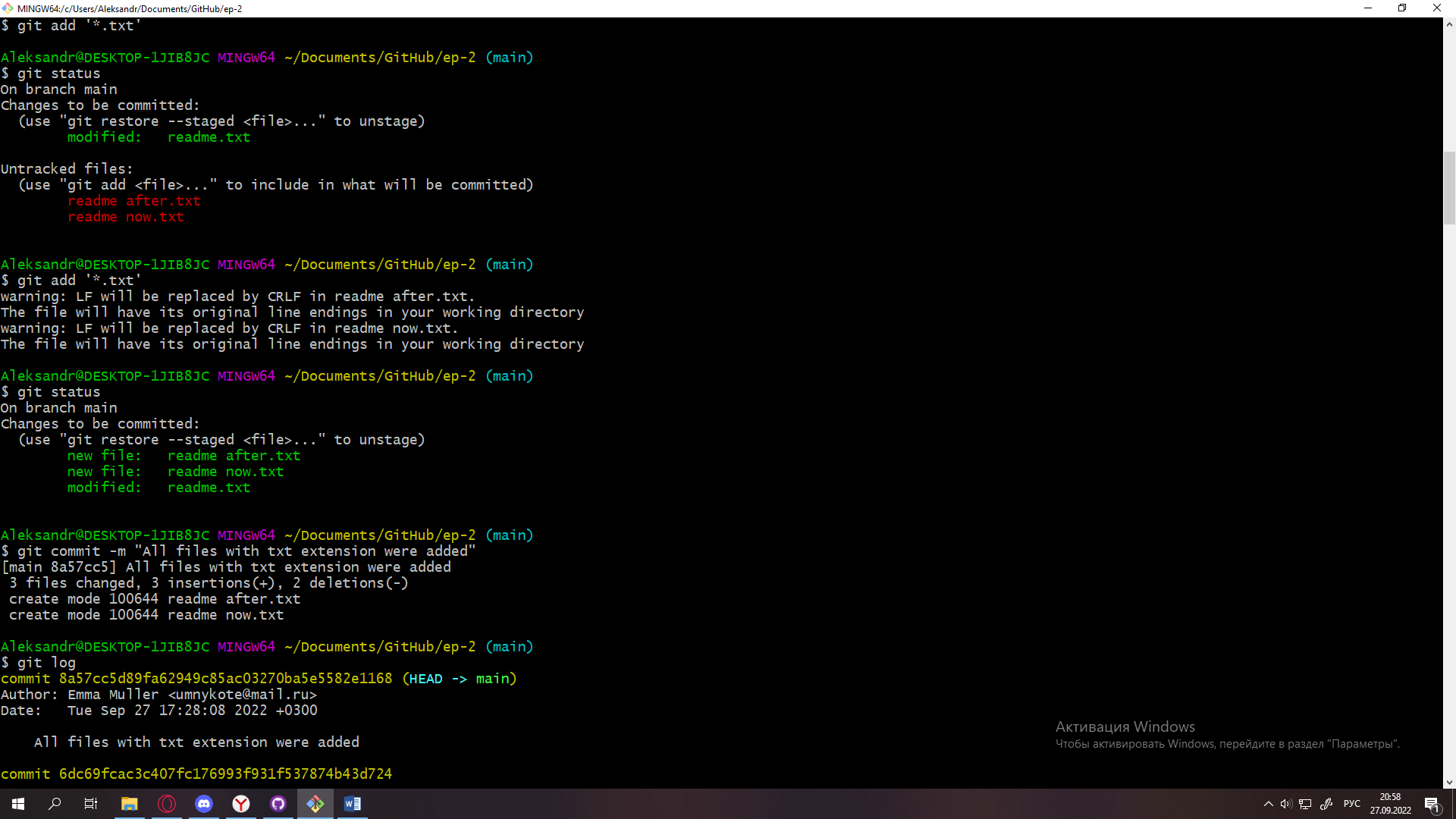


Если мы внесем какую-нибудь новую информацию в зафиксированный файл, то он после этого станет снова не отслеживаемым. Сделаем его вновь отслеживаемым.

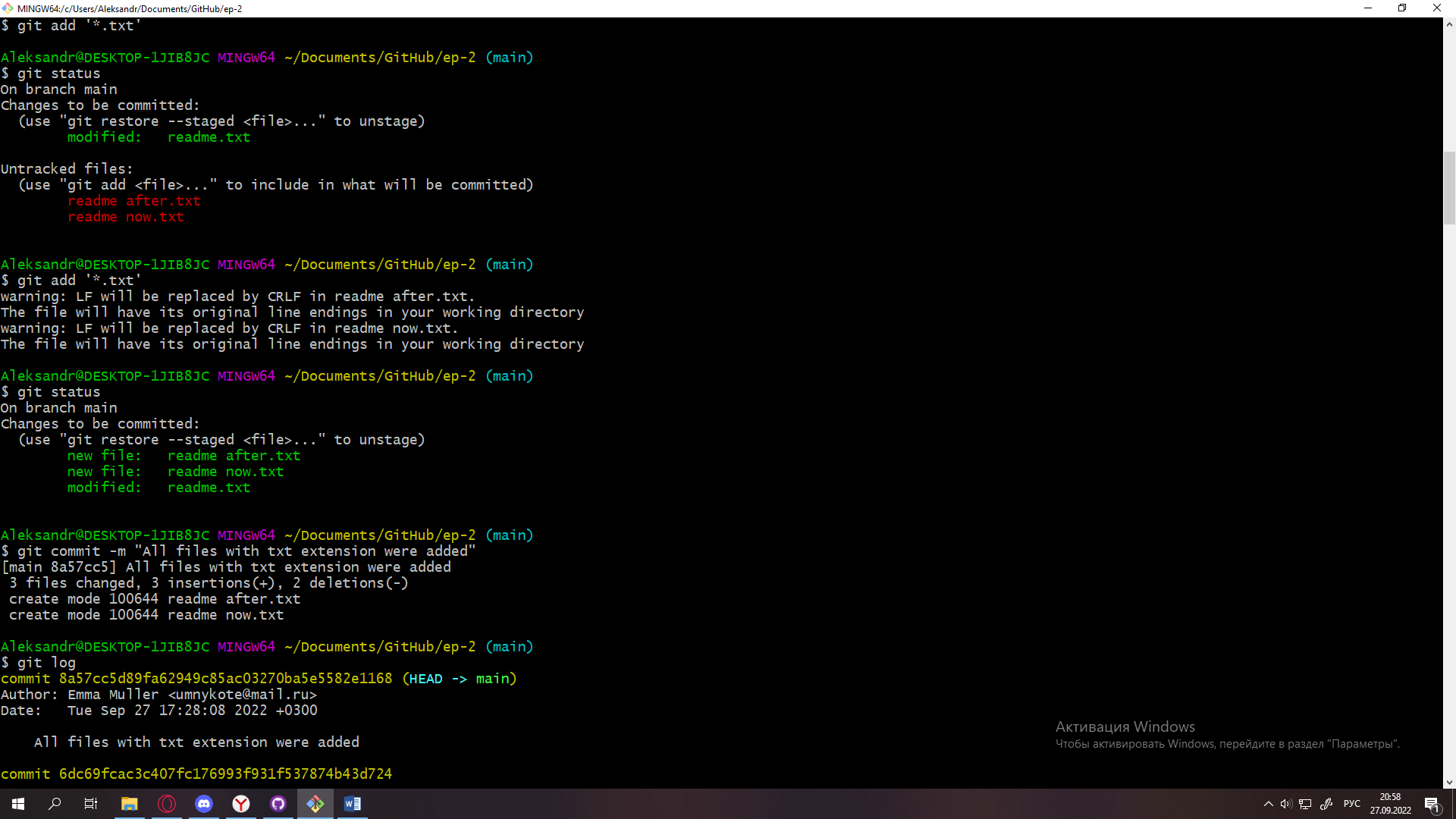
  
Для быстроты и удобства можно добавить маску, по которой можно закоммитить множество файлов. Например:



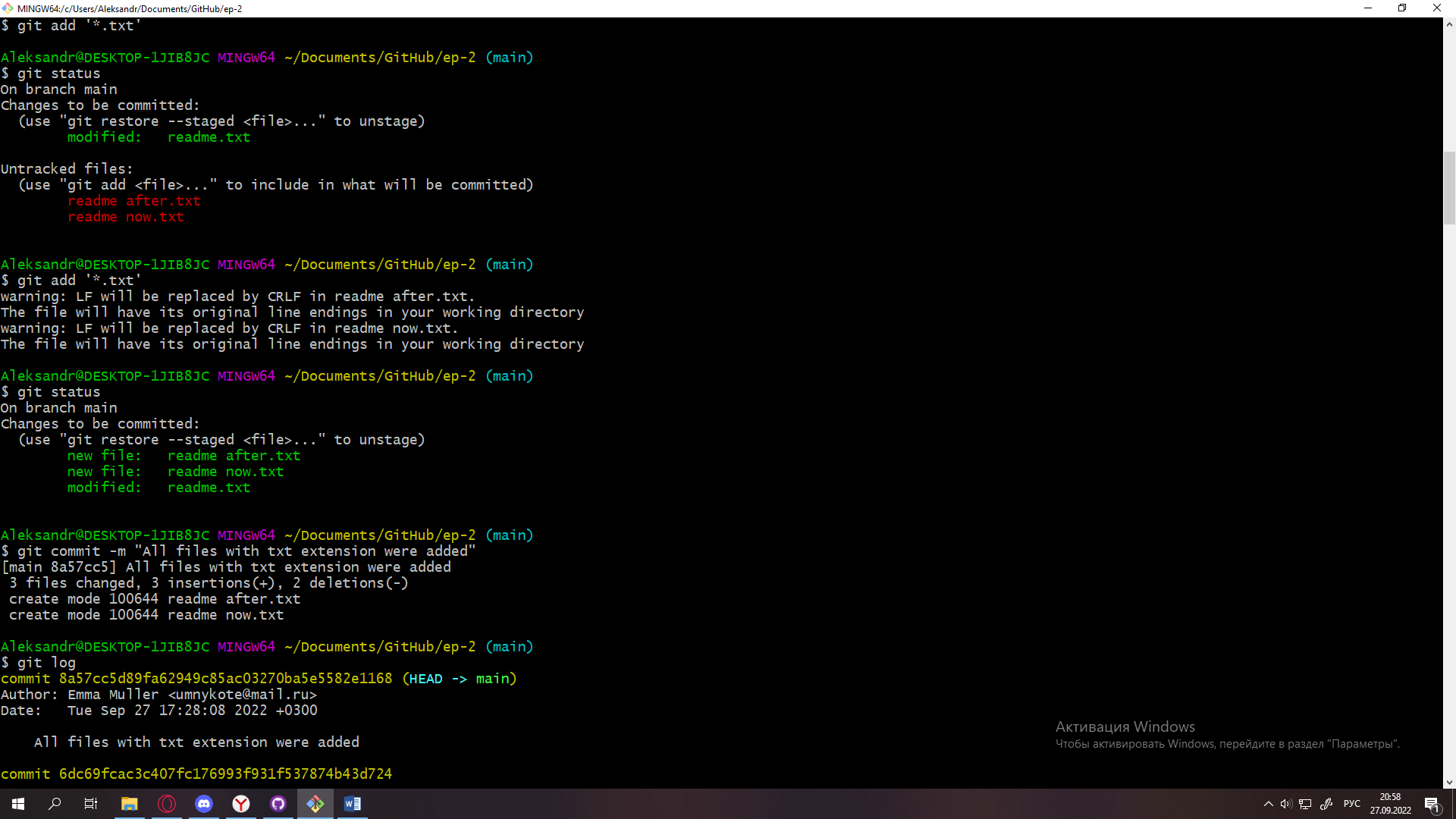
Создадим копии файла readme.txt – readme after.txt, readme now.txt. Можно увидеть, что отслеживается лишь исходный файл, а две его копии – нет.



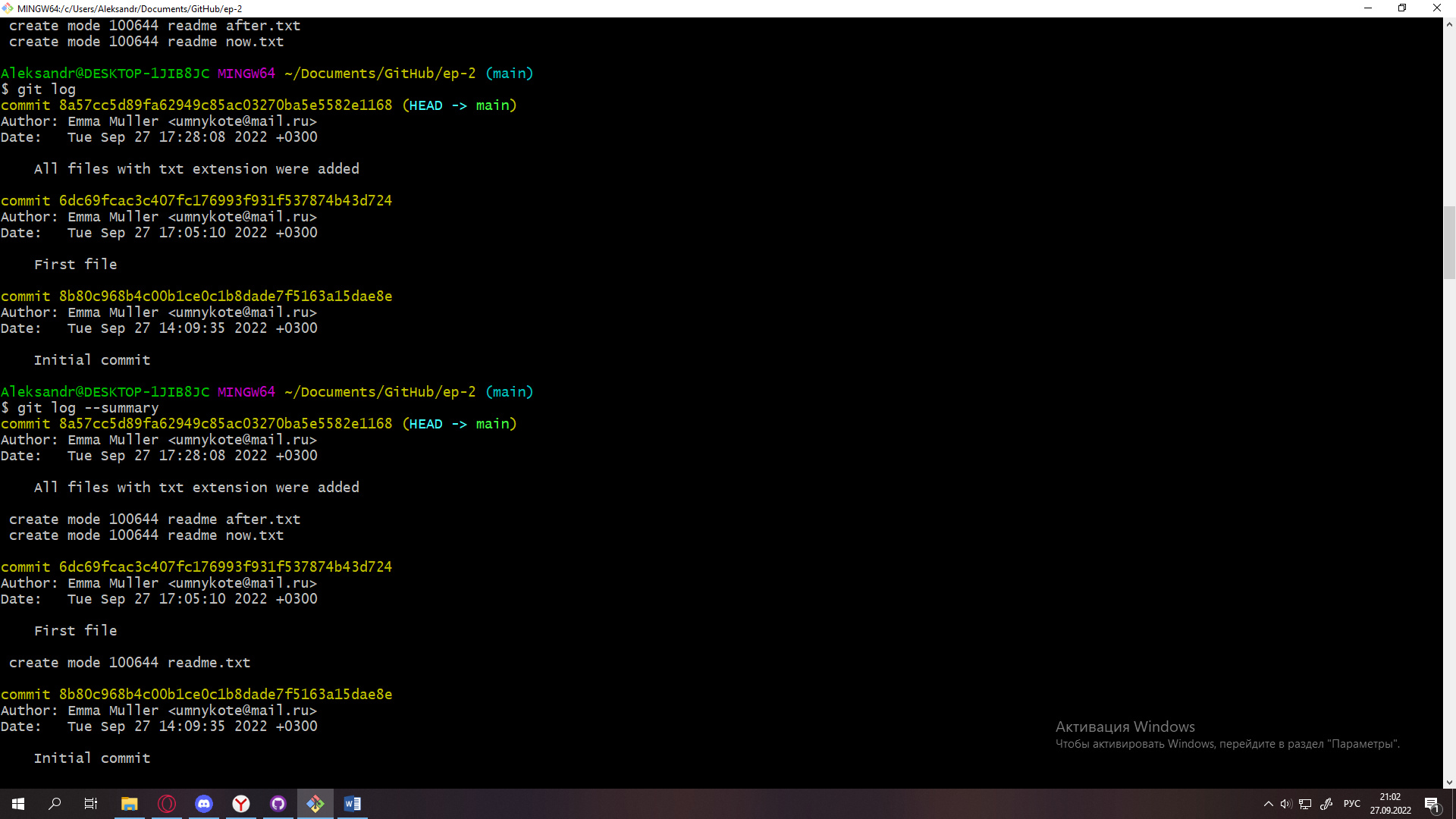
Исправим это, сделав все файлы отслеживаемыми. Используем маску.



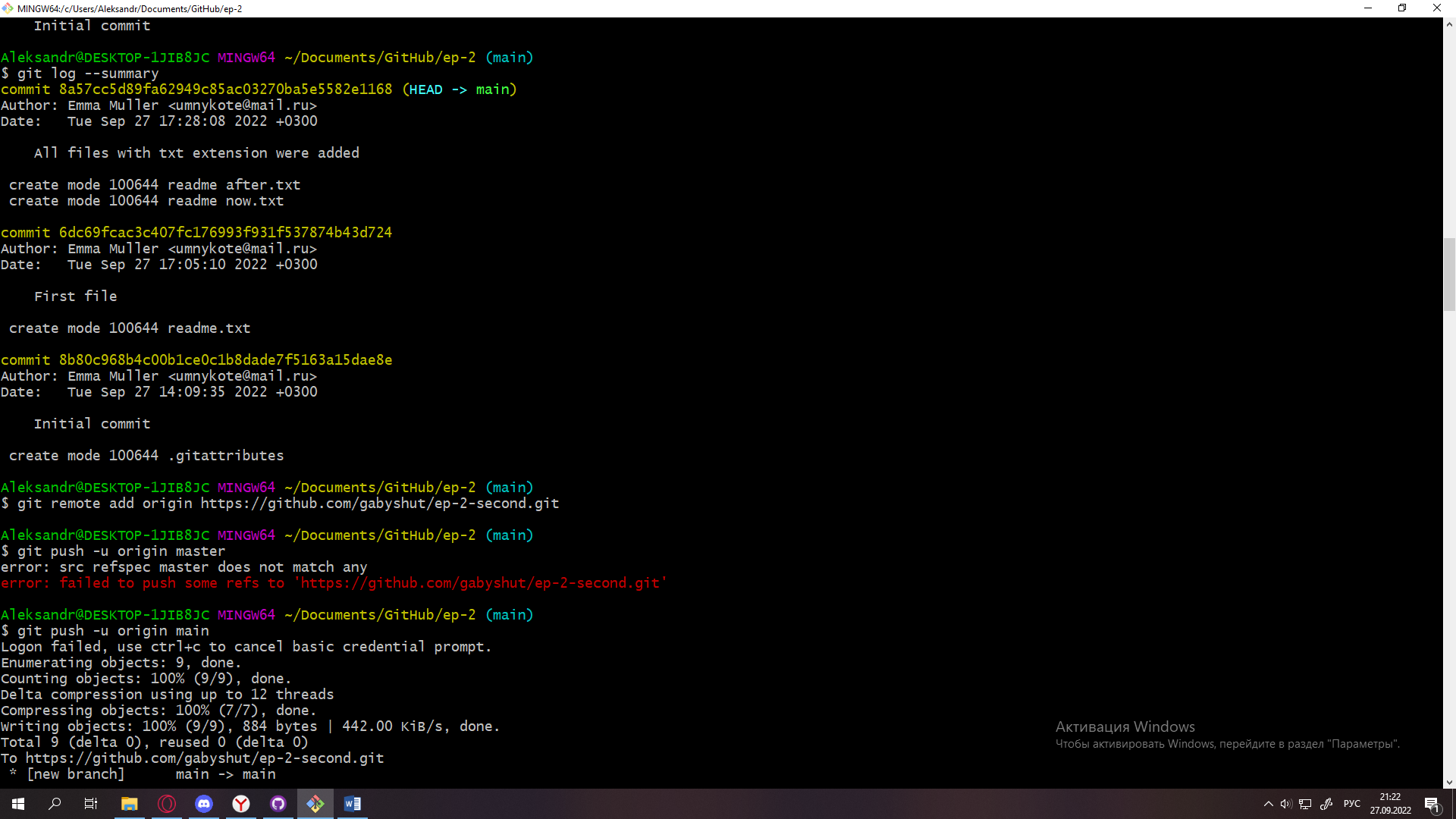
Теперь все файлы отслеживаются. Зафиксируем все файлы.



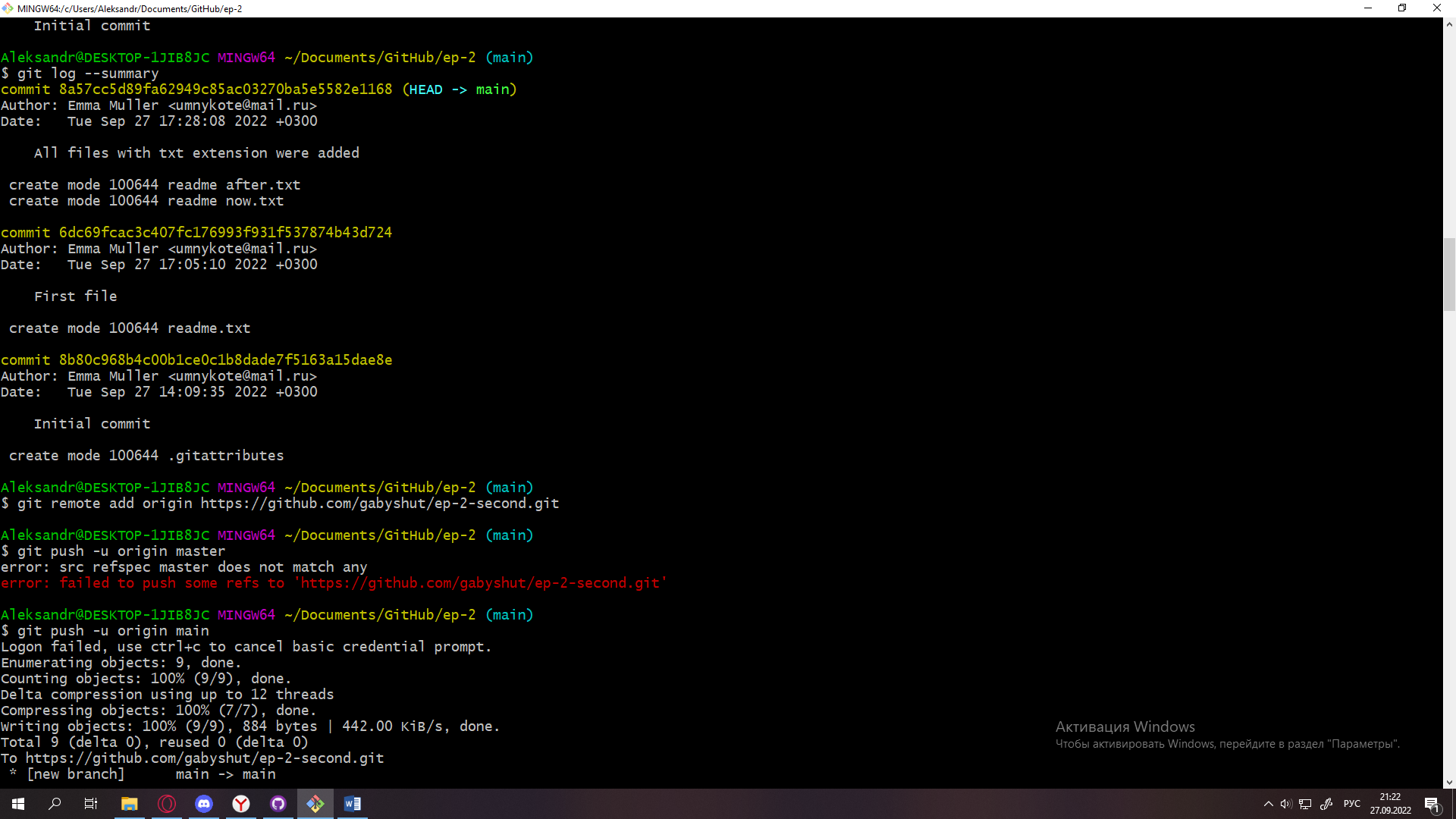
Для того, чтобы посмотреть историю всех коммитов (от самого нового, до самого старого), необходимо использовать команду git log.



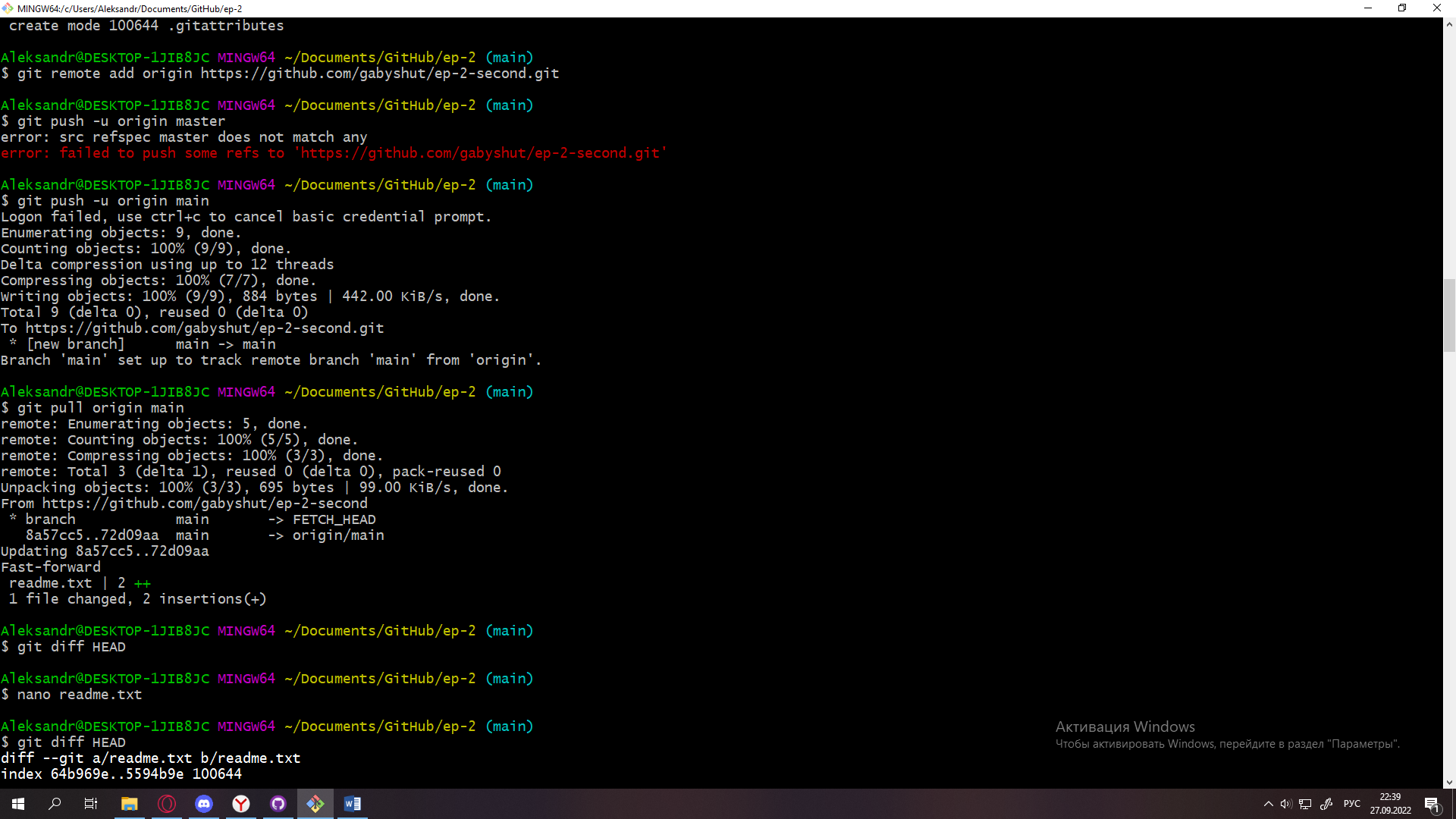
С помощью команды git log --summary на экран выведется больше информации, чем при git log.



Добавление удаленного репозитория. Чтобы добавить удаленный репозиторий, необходимо написать команду git remote add [название (чаще всего origin)] [ссылка на репозиторий]. Я заранее создала репозиторий, поэтому имею готовую ссылку.

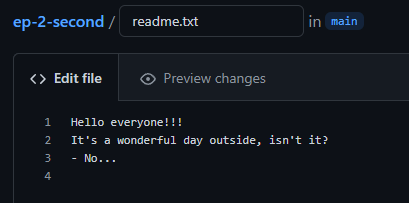


После этого будет предложено ввести имя пользователя и специальный ключ, который можно создать в настройках аккаунта (я делала на GitHub). После ввода ключа создается удаленный репозиторий, связанный с созданным на платформе GitHub. Импортируем все файлы репозитория «ep-2» в удаленный репозиторий с помощью команды git push –u [название удаленного репозитория] main (master).

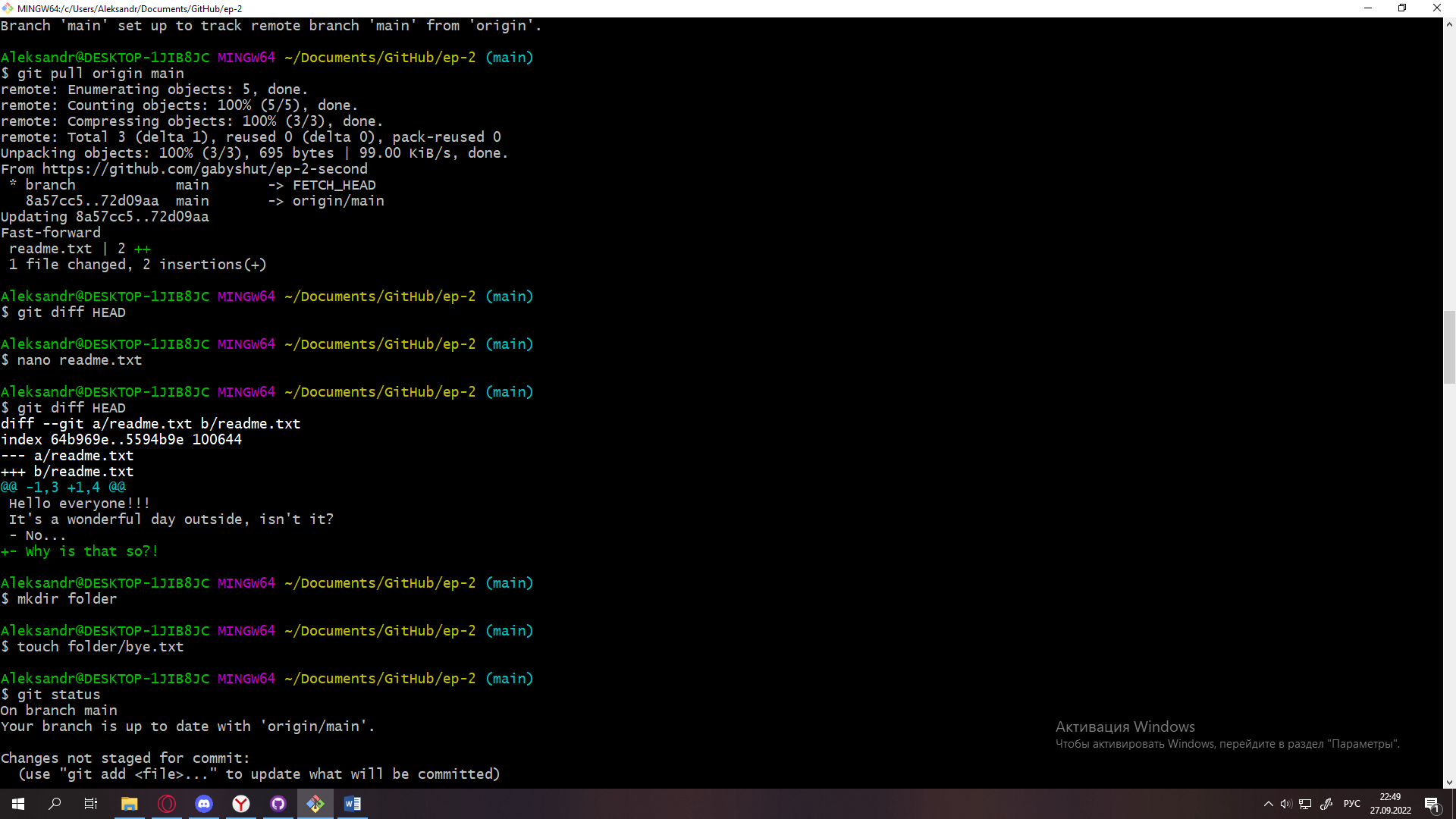


После этого в созданном на GitHub репозитории появятся все созданные ранее файлы, хранящиеся в репозитории «ep-2». Они доступны для просмотра, если репозиторий публичный.

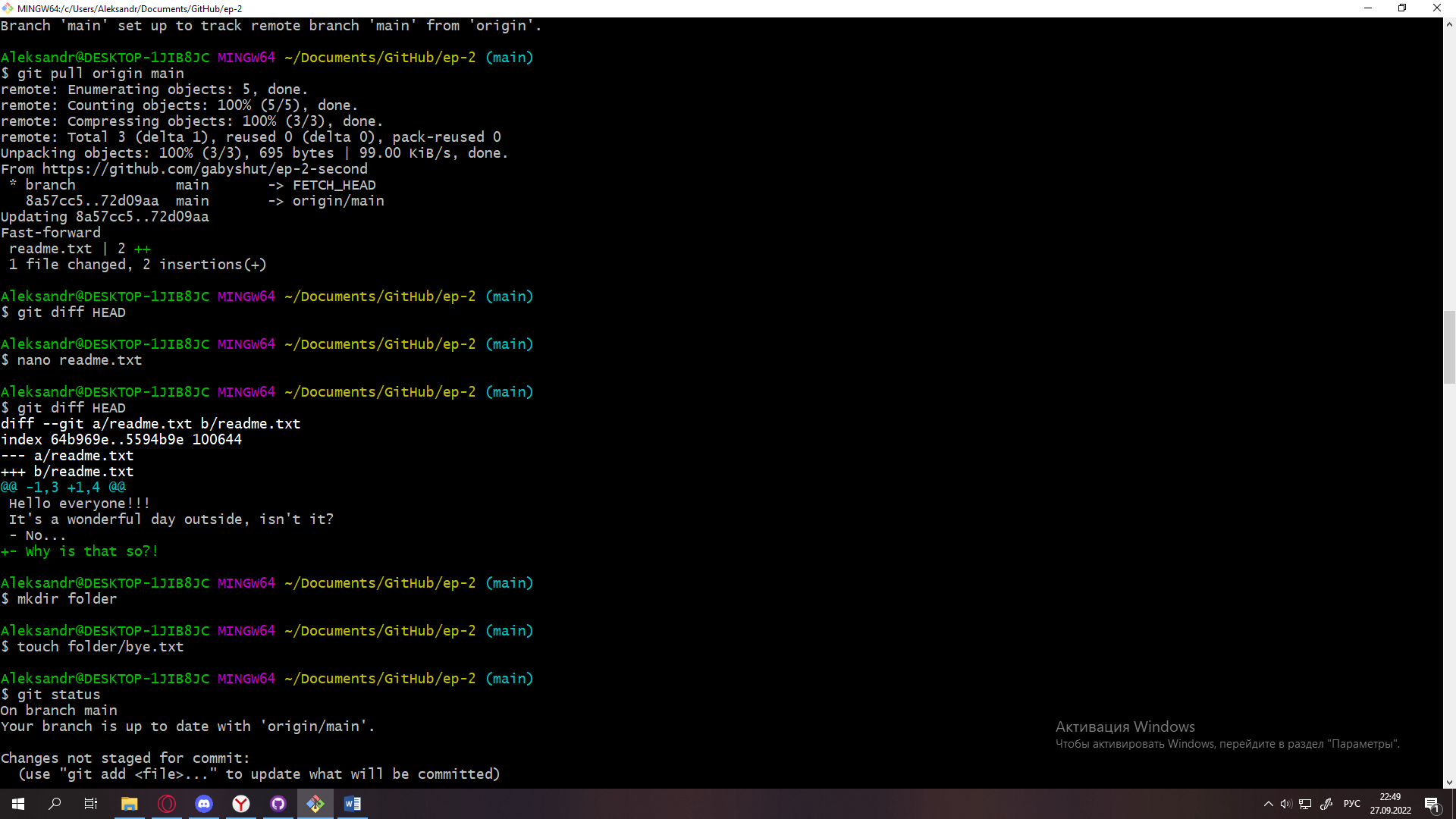
Внесем изменения в файле readme.txt в удаленном репозитории.



Чтобы изменения также были произведены и в репозитории «ep-2» необходимо использовать команду git pull origin main (master).

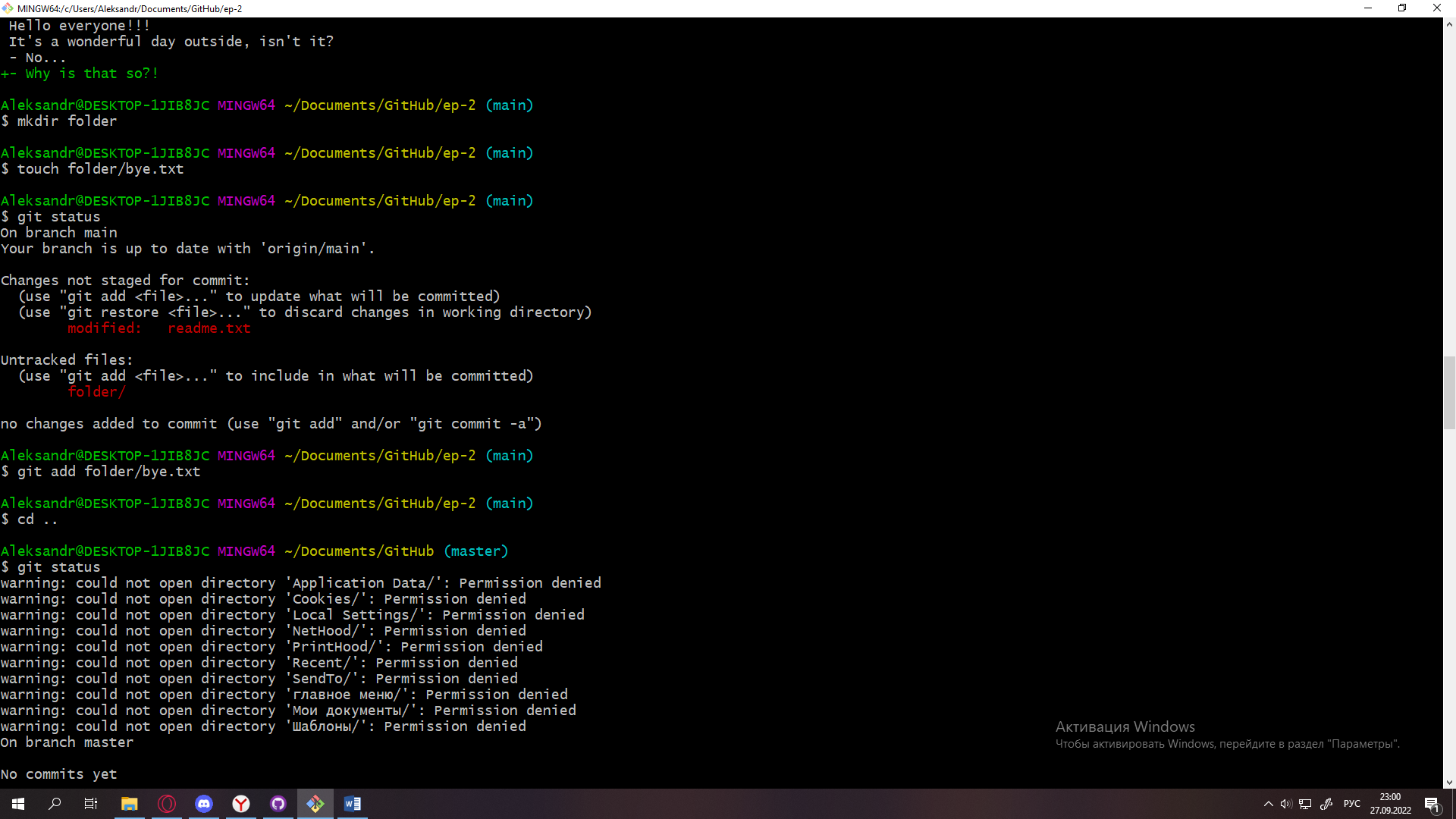


Чтобы сделать какие-либо изменения в файле без лишних коммитов, необходимо ввести git diff HEAD и редактировать файл. Например:

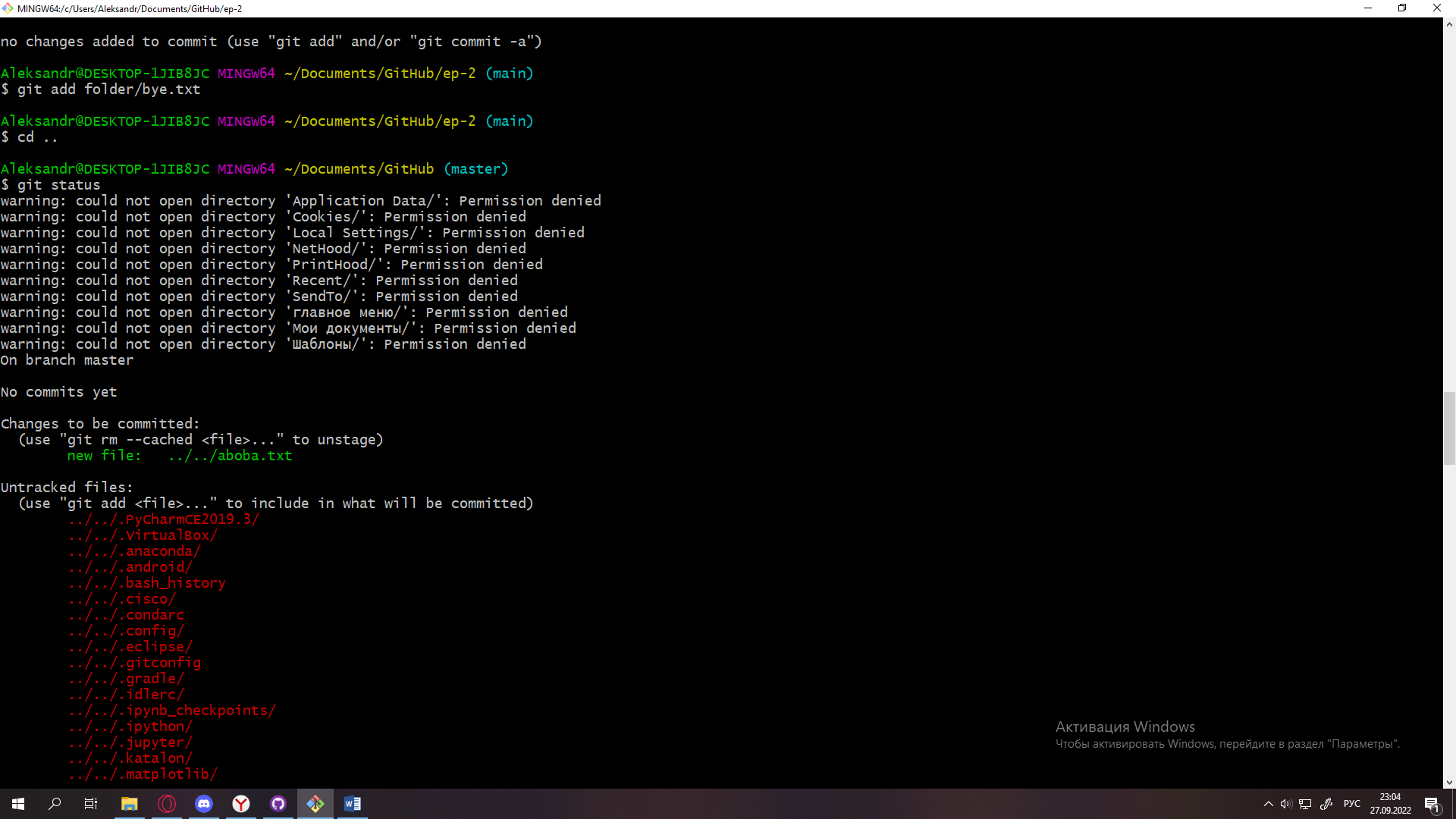


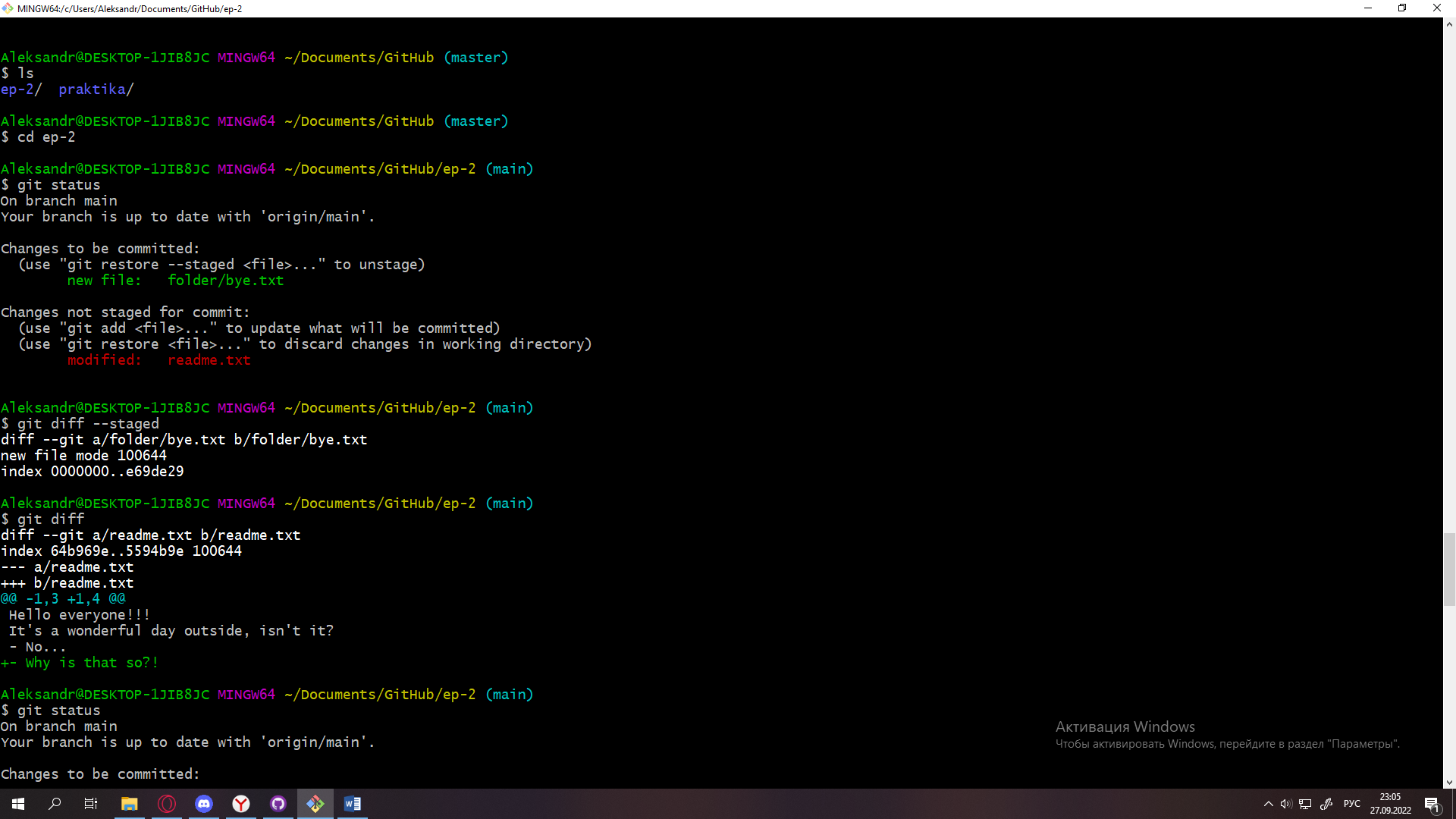
HEAD – указатель на последнюю зафиксированную версию.

Создадим каталог (с помощью команды mkdir) и файл внутри него. Можно заметить, что каталог не отслеживается.



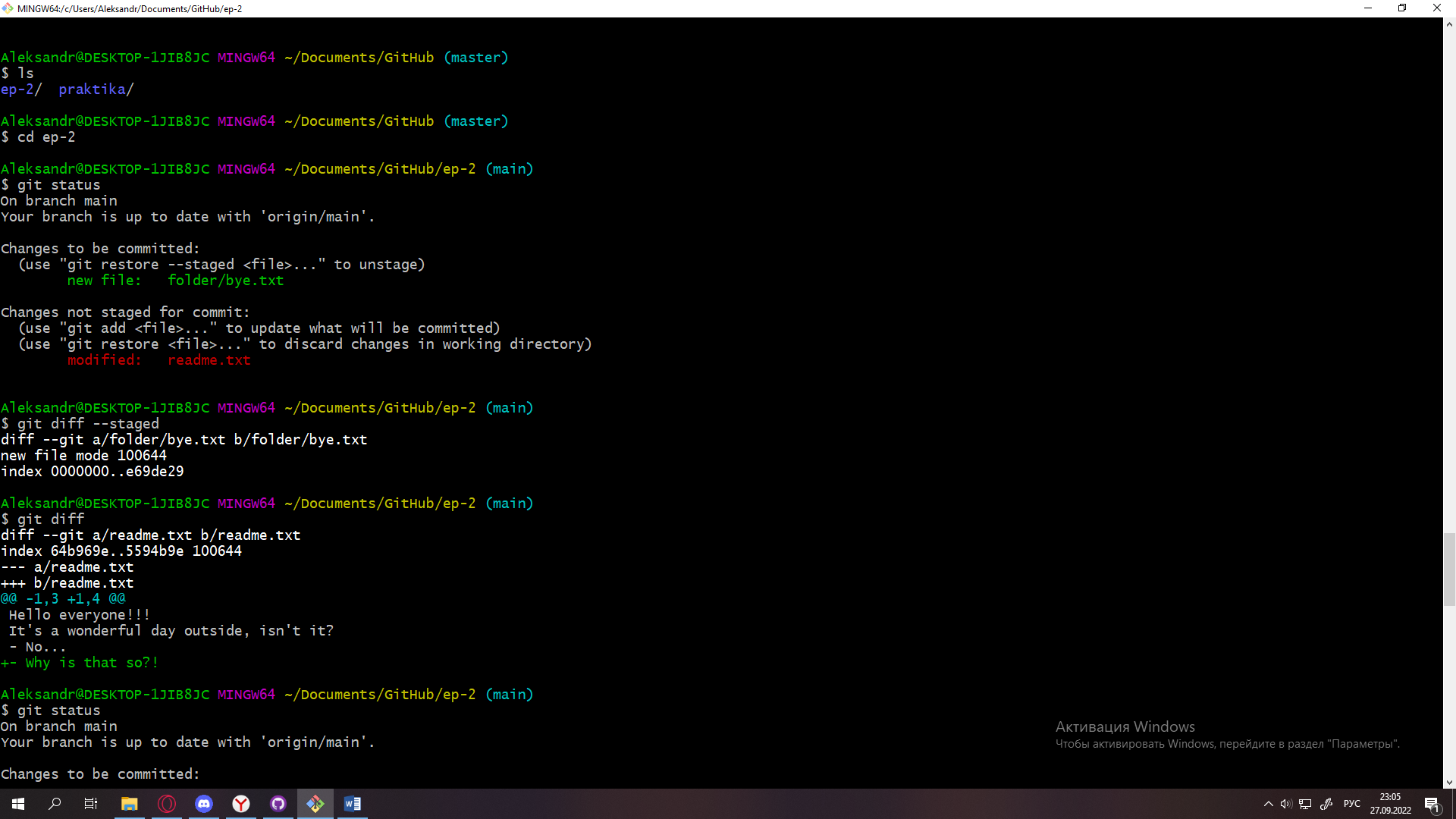
Чтобы отслеживать данный каталог, необходимо произвести такой ряд действий:





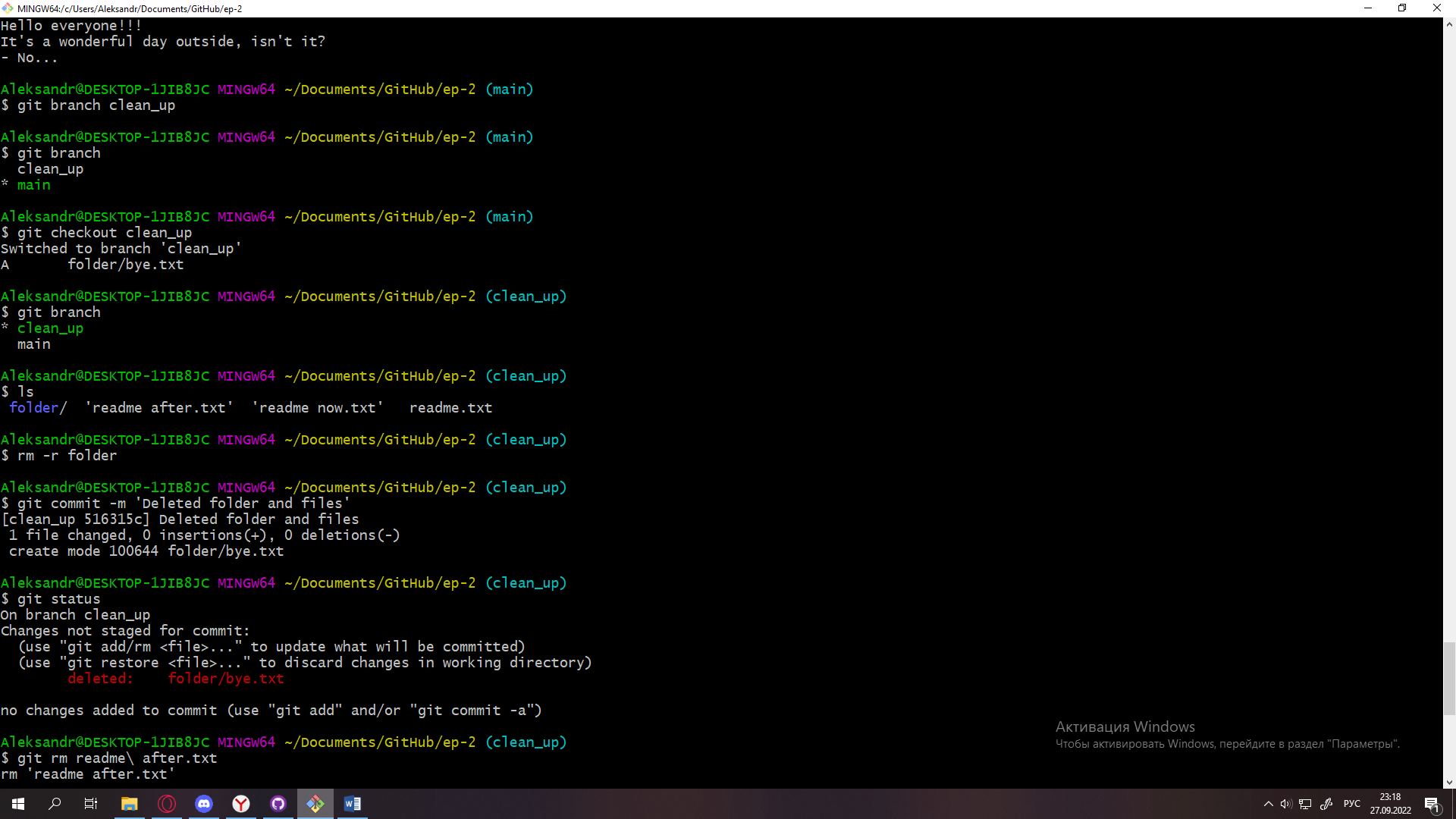
Теперь каталог отслеживается.

Чтобы сравнить состояние репозитория с тем, которое было, необходимо выполнить операцию git diff --staged. Эта команда позволяет посмотреть на те изменения, которые были сделаны в стадии staged.

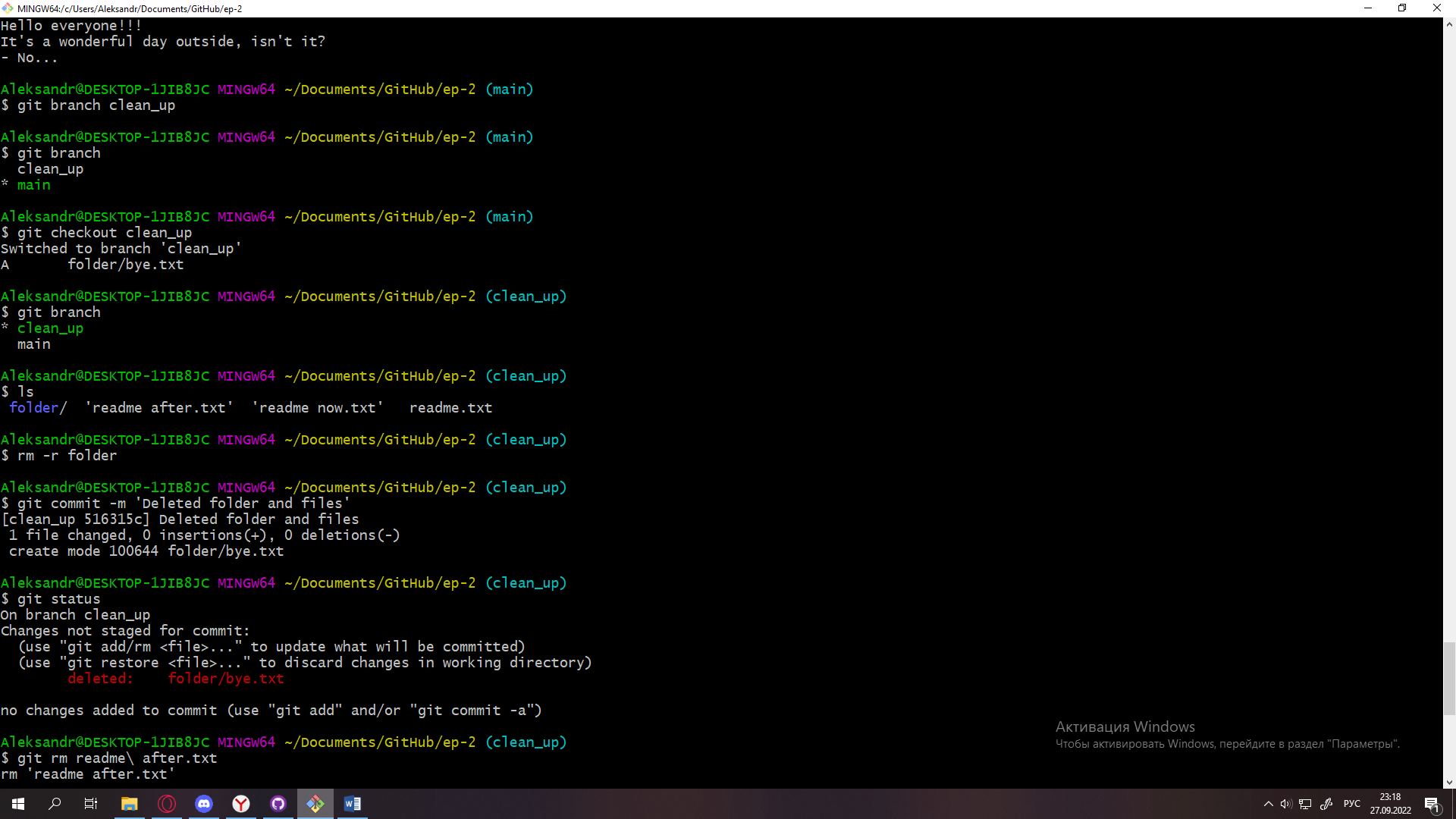


Если мы хотим отменить это действие мы можем выполнить операцию git reset (folder/bye.txt).

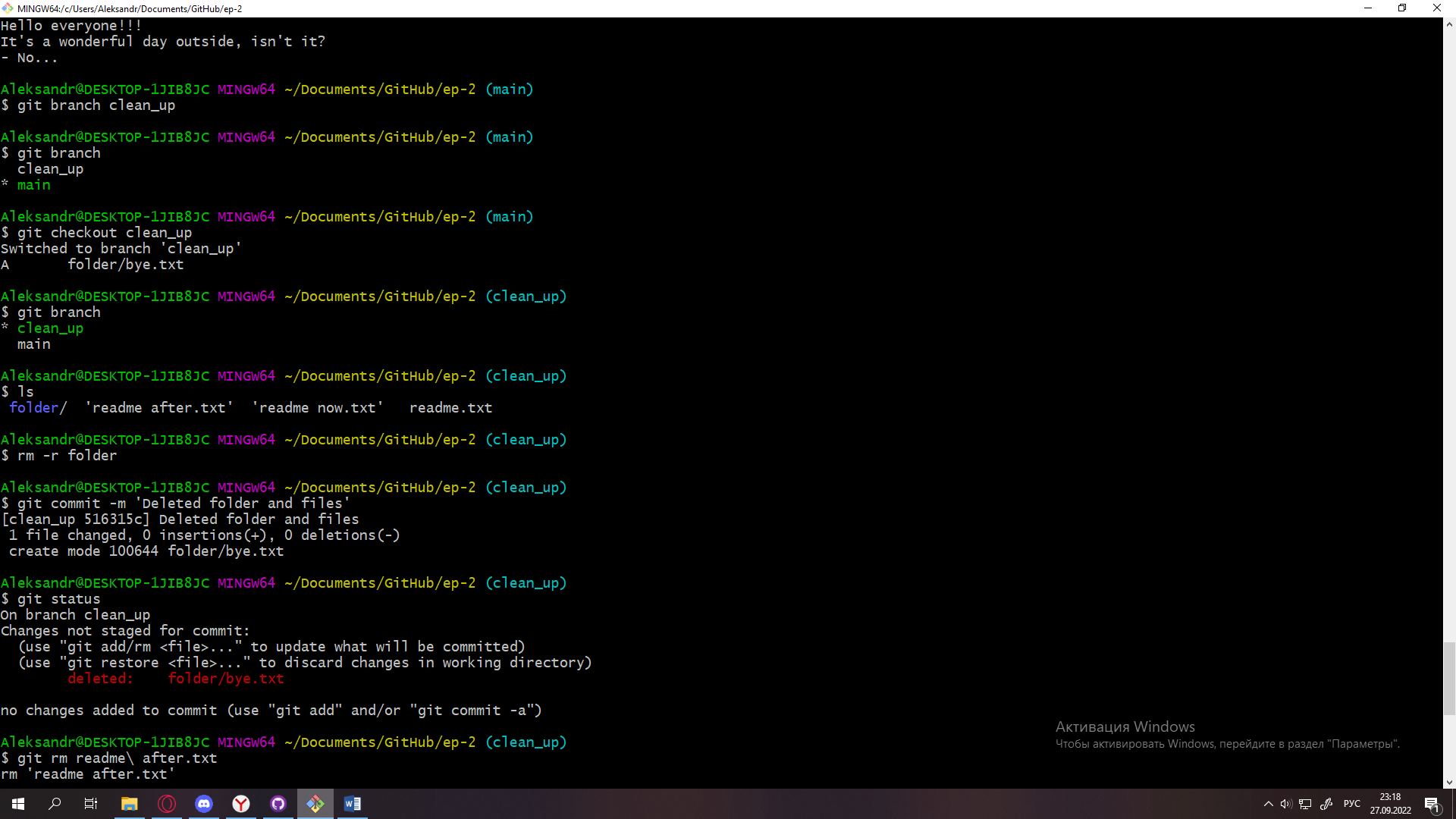
Чтобы создать ветку, необходимо ввести команду git branch [название]. Чтобы посмотреть, в какой ветке мы работаем, введем команду git branch.



Чтобы переключиться на другую ветку, надо ввести команду git checkout [название ветки].



Удалю каталог folder. Для этого введу команду rm –r folder.



Основные команды были рассмотрены.