Шарипа В.Г. 2-4А

**ОТЧЁТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ**

**Изучение функционала GitHub**

**Git** — это набор консольных утилит, которые отслеживают и фиксируют изменения в файлах (чаще всего речь идет об исходном коде программ, но вы можете использовать его для любых файлов на ваш вкус). С его помощью вы можете откатиться на более старую версию вашего проекта, сравнивать, анализировать, сливать изменения и многое другое. Этот процесс называется контролем версий.

**Git** является распределенным, то есть не зависит от одного центрального сервера, на котором хранятся файлы. Вместо этого он работает полностью локально, сохраняя данные в папках на жестком диске, которые называются репозиторием. Тем не менее, вы можете хранить копию репозитория онлайн, это сильно облегчает работу над одним проектом для нескольких людей. Для этого используются сайты вроде github и bitbucket.

**Концепция git:**

* Рабочая директория — файловая система проекта (те файлы, с которыми вы работаете).
* Индекс — список отслеживаемых git-ом файлов и директорий, промежуточное хранилище изменений (редактирование, удаление отслеживаемых файлов).
* Директория .git/ — все данные контроля версий этого проекта (вся история разработки: коммиты, ветки, теги и пр.).
* Коммит — «сохранение» (хранит набор изменений, сделанный в рабочей директории с момента предыдущего коммита). Коммит неизменен, его нельзя отредактировать.У всех коммитов (кроме самого первого) есть один или более родительских коммитов, поскольку коммиты хранят изменения от предыдущих состояний.

**Указатели:**

* HEAD — указатель на текущий коммит или на текущую ветку (то есть, в любом случае, на коммит). Указывает на родителя коммита, который будет создан следующим.
* ORIG\_HEAD — указатель на коммит, с которого вы только что переместили HEAD (командой git reset ..., например).
* Ветка (master, develop etc.) — указатель на коммит. При добавлении коммита, указатель ветки перемещается с родительского коммита на новый.
* Теги — простые указатели на коммиты. Не перемещаются.

**Основные команды Git:**

**VIM - команды:**

# Нажатия кнопок

ESC — переход в командный режим

i — переход в режим редактирования текста

ZQ (зажат Shift, поочередное нажатие) — выход без сохранения

ZZ (зажат Shift, поочередное нажатие) — сохранить и выйти

```bash

# Нажатия кнопок

ESC — переход в командный режим

i — переход в режим редактирования текста

ZQ (зажат Shift, поочередное нажатие) — выход без сохранения

ZZ (зажат Shift, поочередное нажатие) — сохранить и выйти

# Ввод в командном режиме

:q! — выйти без сохранения

:wq — сохранить файл и выйти

:w filename.txt — сохранить файл как filename.txt

**Консольные команды:**

git init # создать новый проект в текущей директории

git init folder-name # создать новый проект в указанной директории

git status # показать состояние репозитория (отслеживаемые, изменённые, новые файлы и пр.)

git add . # добавить в индекс все новые, изменённые, удалённые файлы из текущей директории и её поддиректорий

git reset # убрать из индекса все добавленные в него изменения (в рабочей директории все изменения сохранятся), антипод git add

git commit -m "Name of commit" # зафиксировать в коммите проиндексированные изменения (закоммитить), добавить сообщение

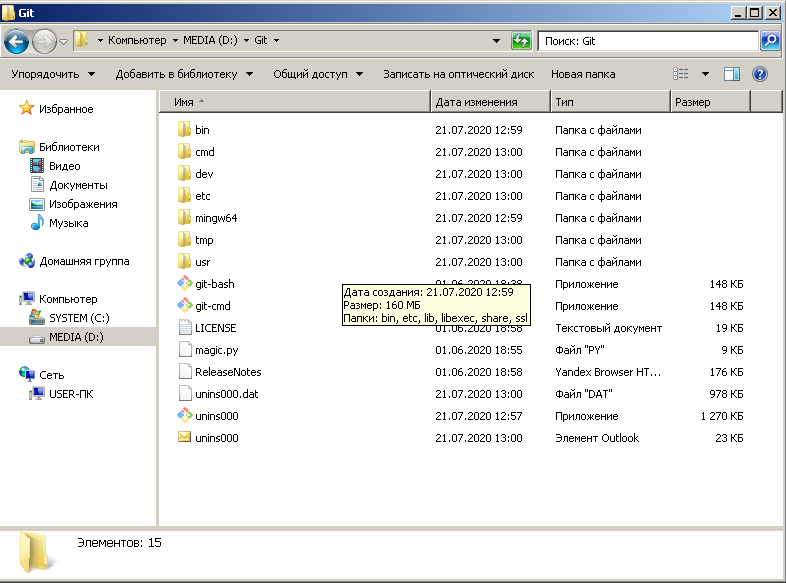
git log master # показать коммиты в указанной ветке

git branch # показать список веток

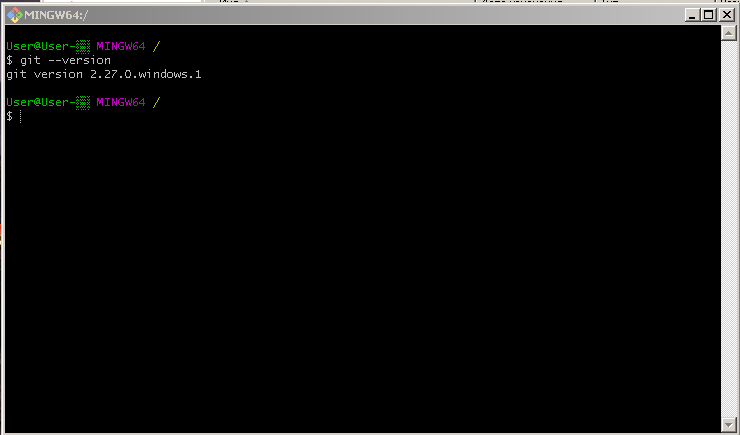
git branch new\_branch # создать новую ветку с указанным именем на текущем коммите

git stash # временно сохранить незакоммиченные изменения и убрать их из рабочей директории

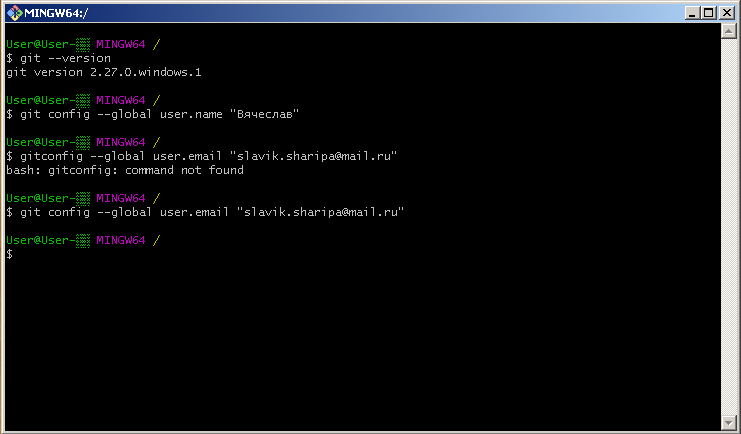
**Ход работы:**

****

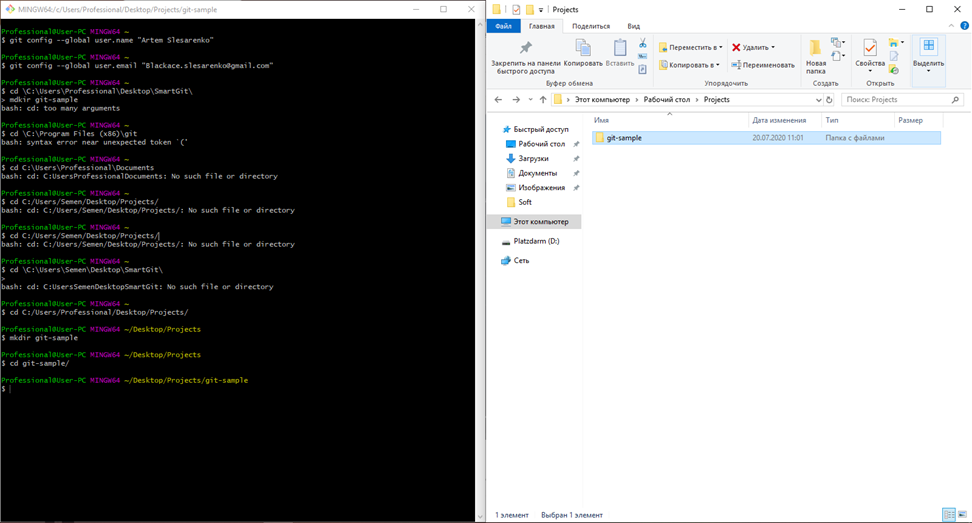
*Установка клиента GIT*

****

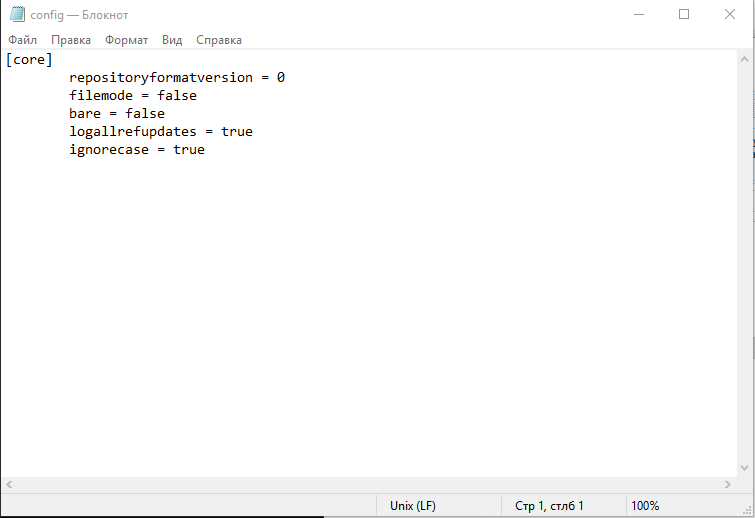
*Рис.1 - Окно программы GIT*

**

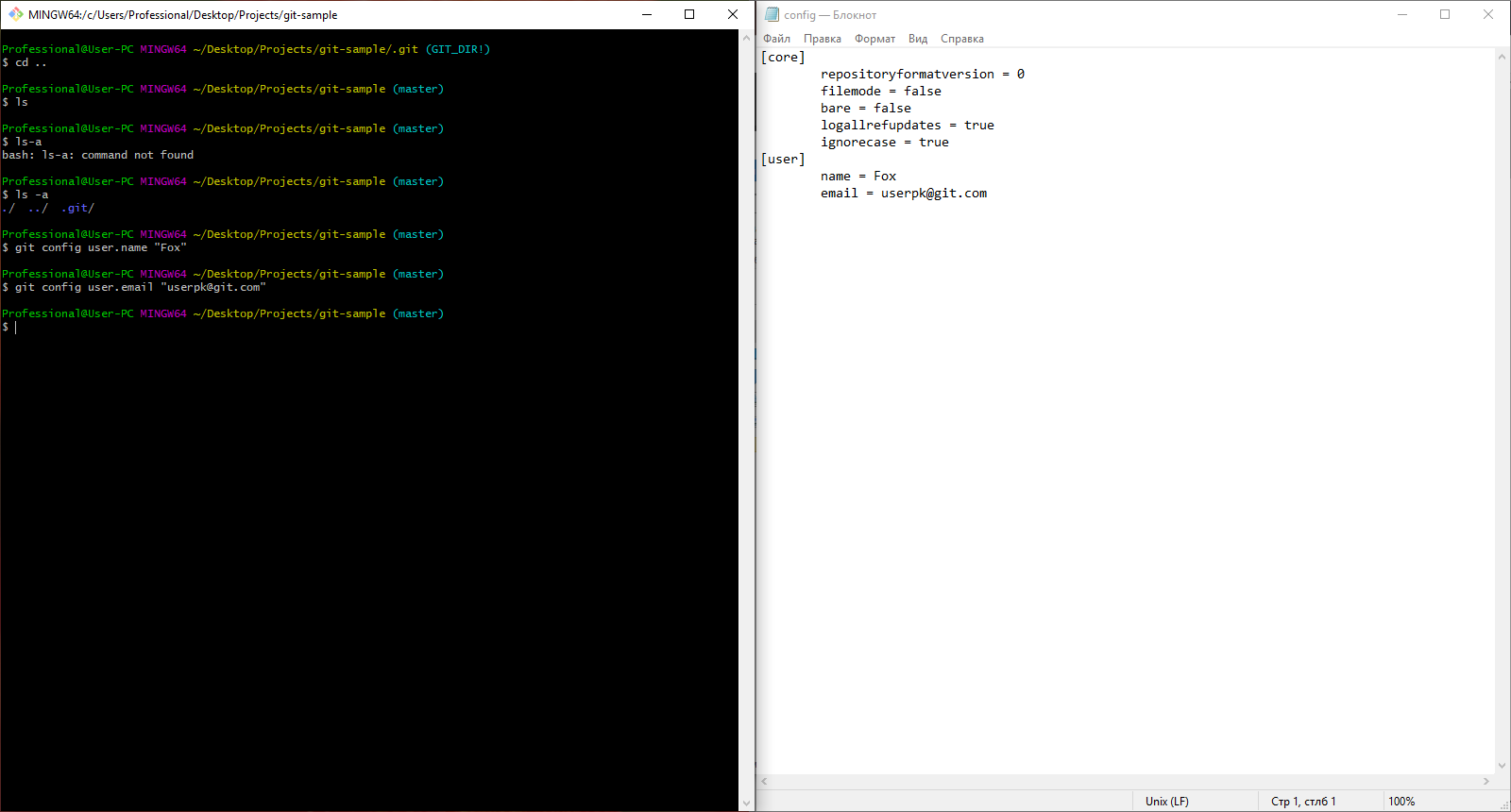
*Рис.2 – Ввод глобальных имени пользователя и электронного адреса*

**

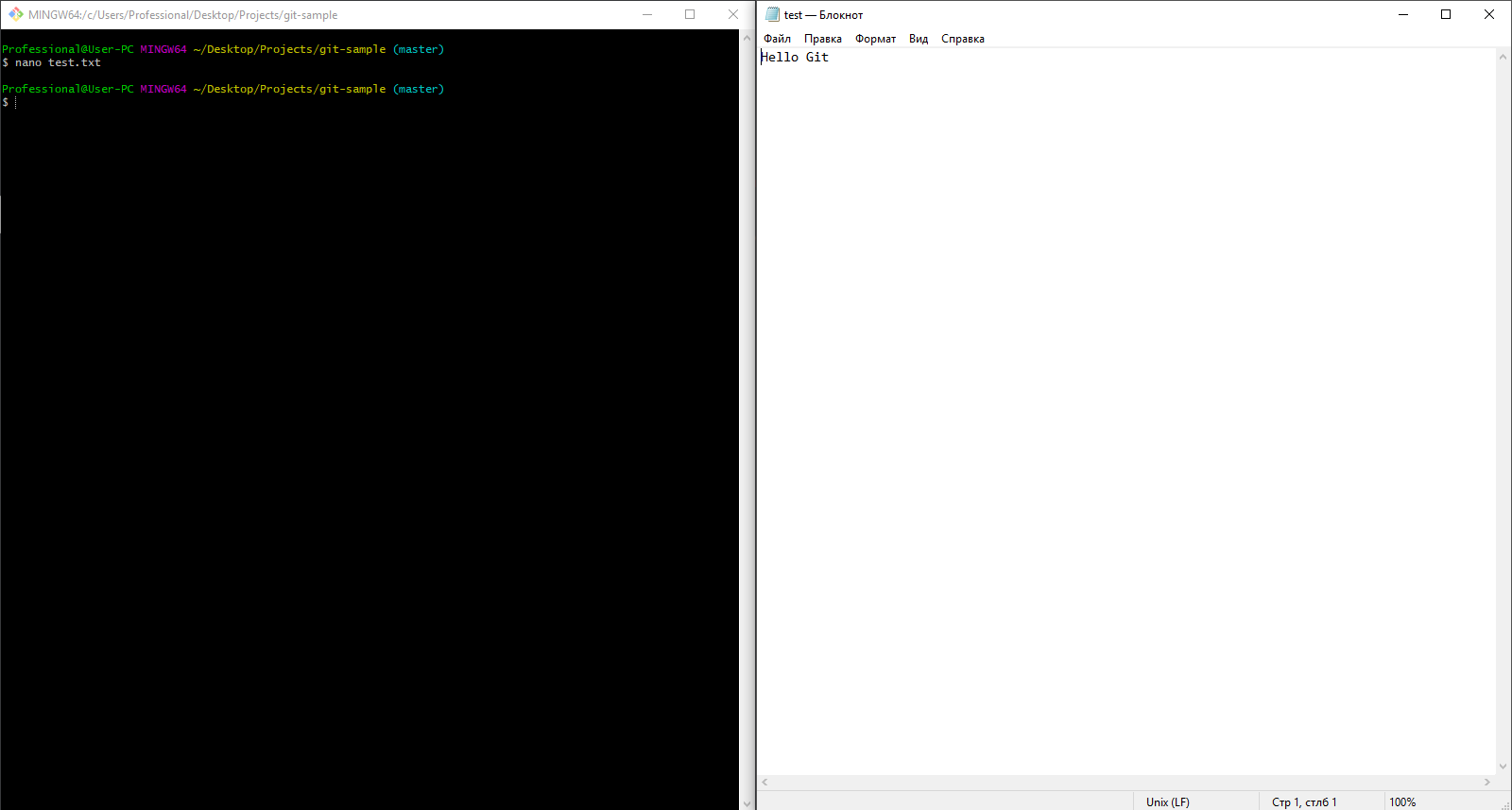
*Рис.3 -Создание рабочей директории*

**

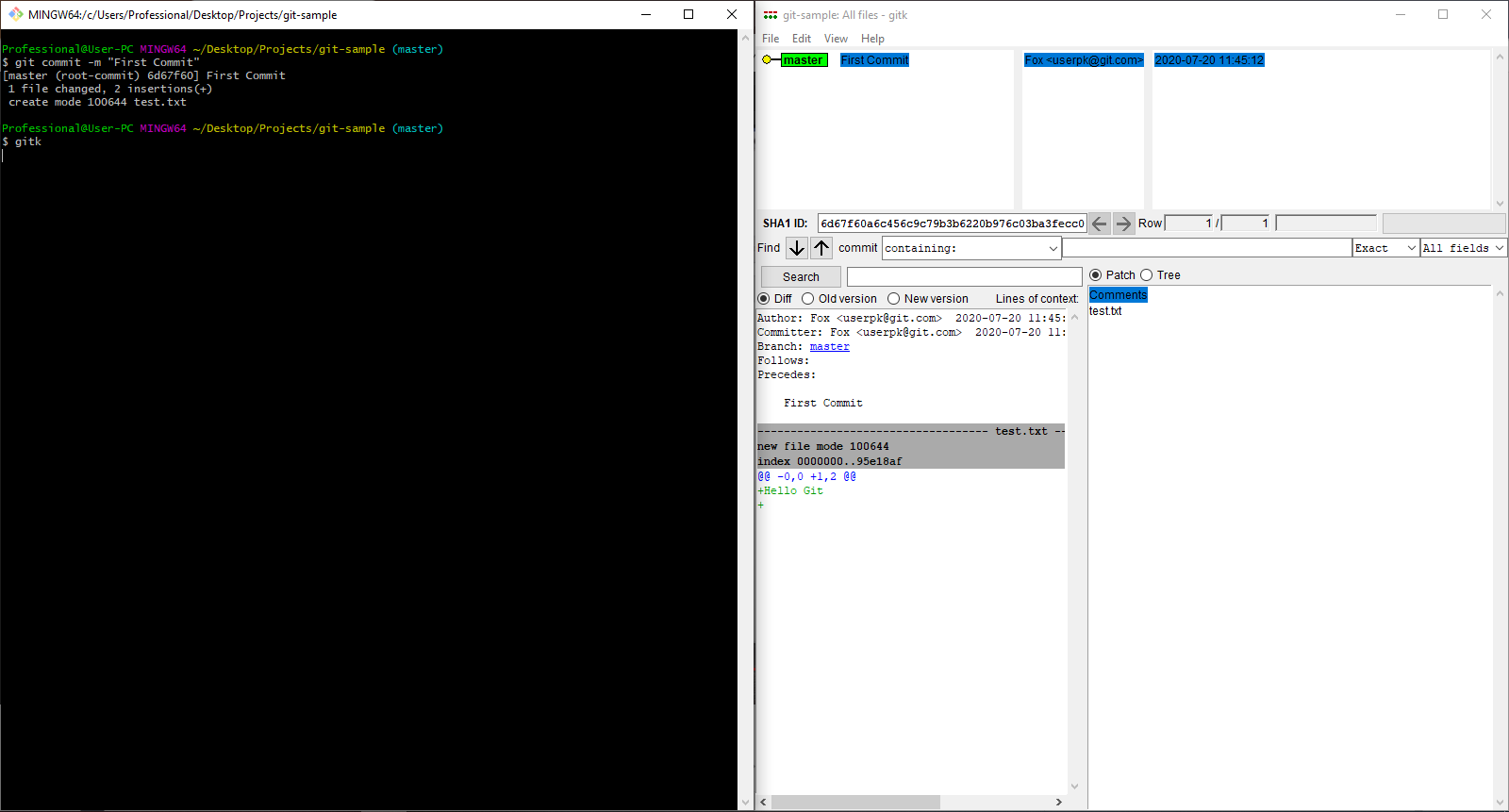
*Рис.4 – Файл Config*



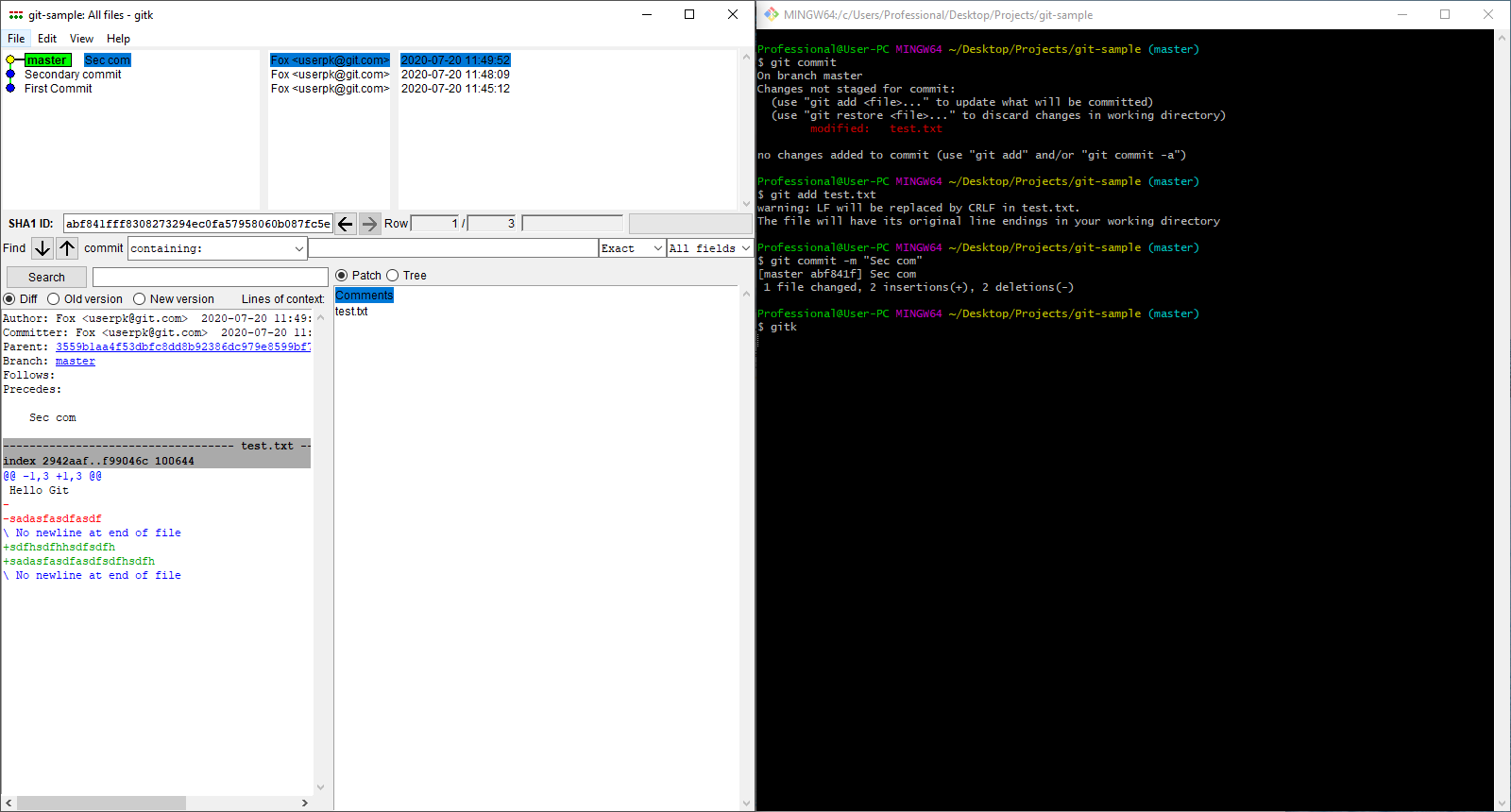
*Рис.5 – Ввод локальных имени пользователя и электронного адреса*

**

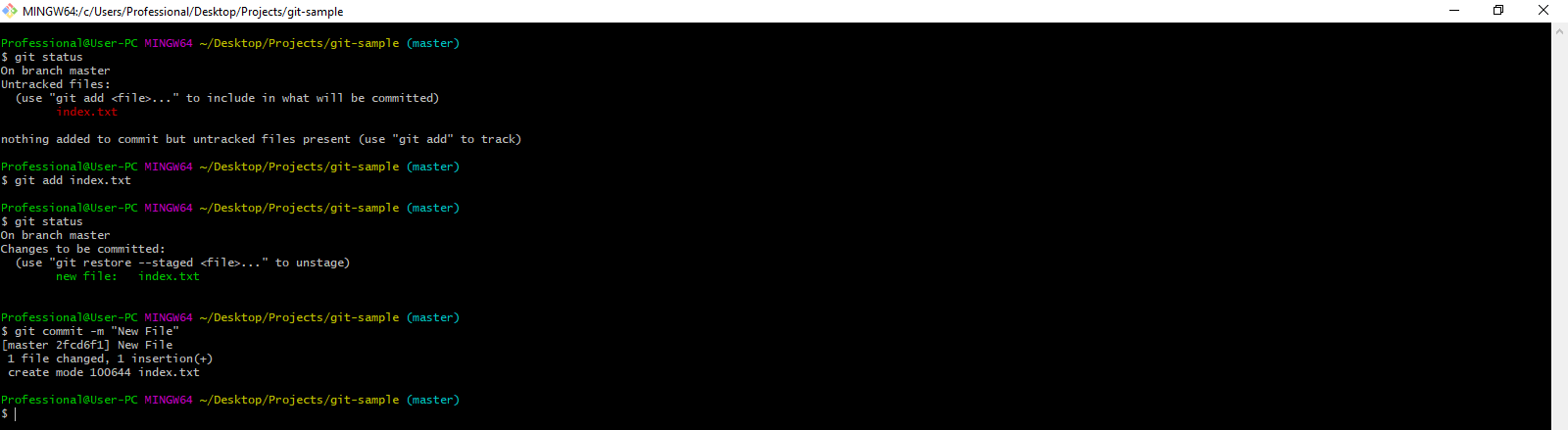
*Рис.6 – Создание текстового файла test.txt*

**

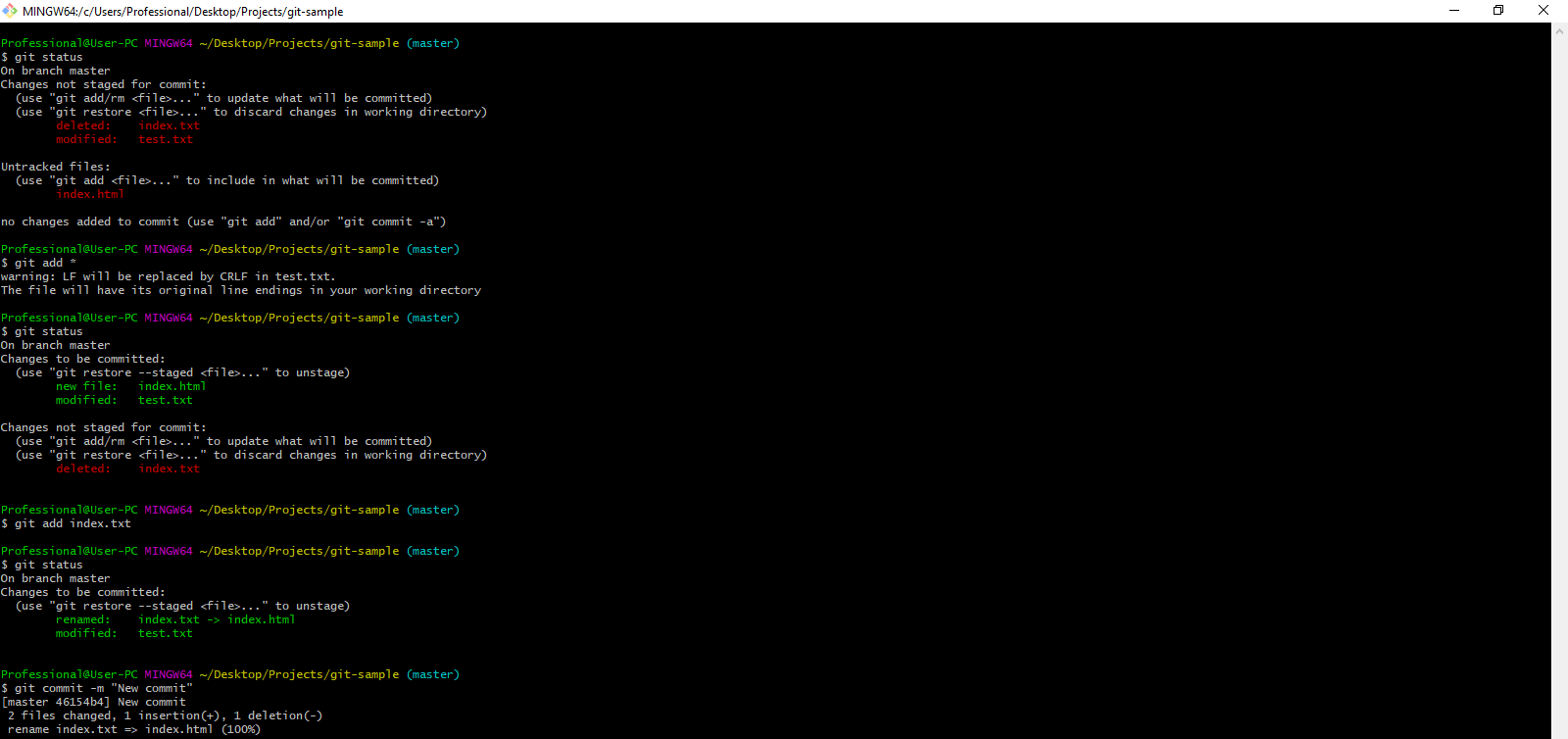
*Рис.7 – Демонстрация команды gitk*

**

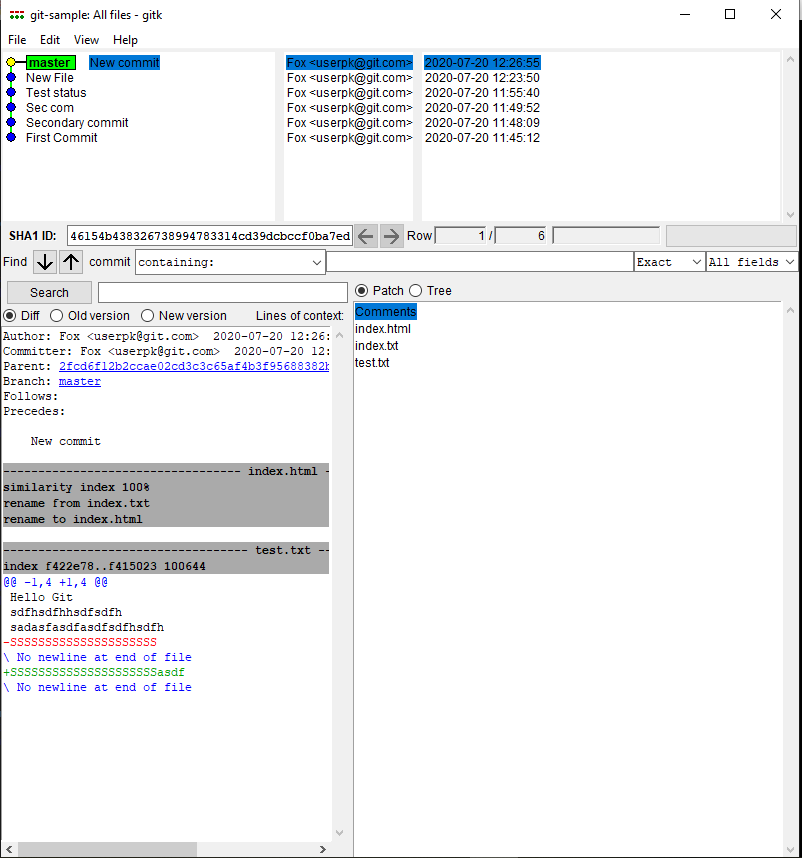
*Рис.8 – Редактирование текстового файла test.txt*

**

*Рис.9 – Создание текстового файла index.txt*

**

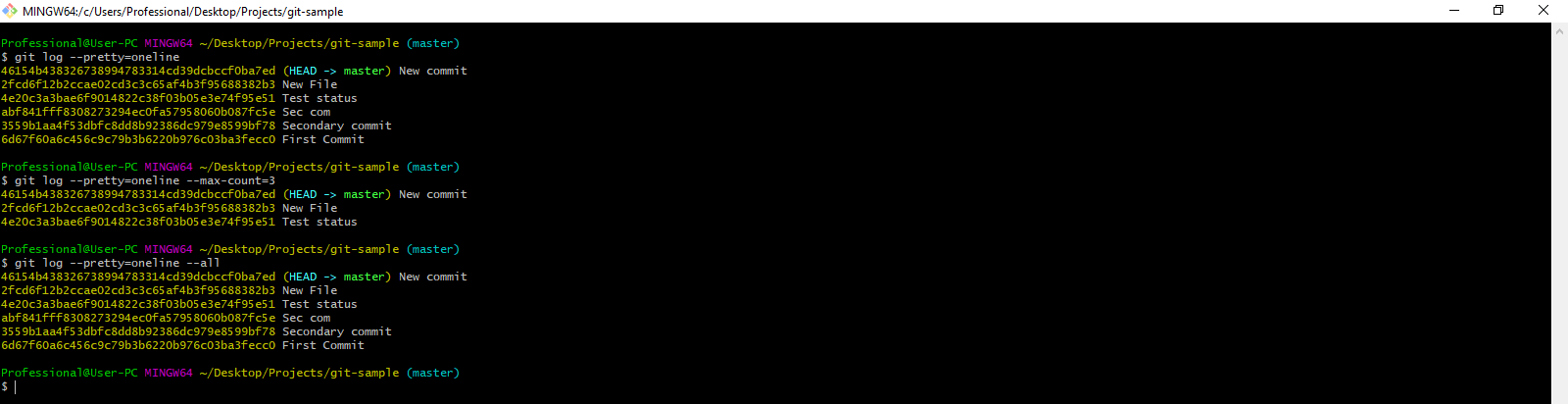
*Рис.10 – Изменение формата файла index*

**

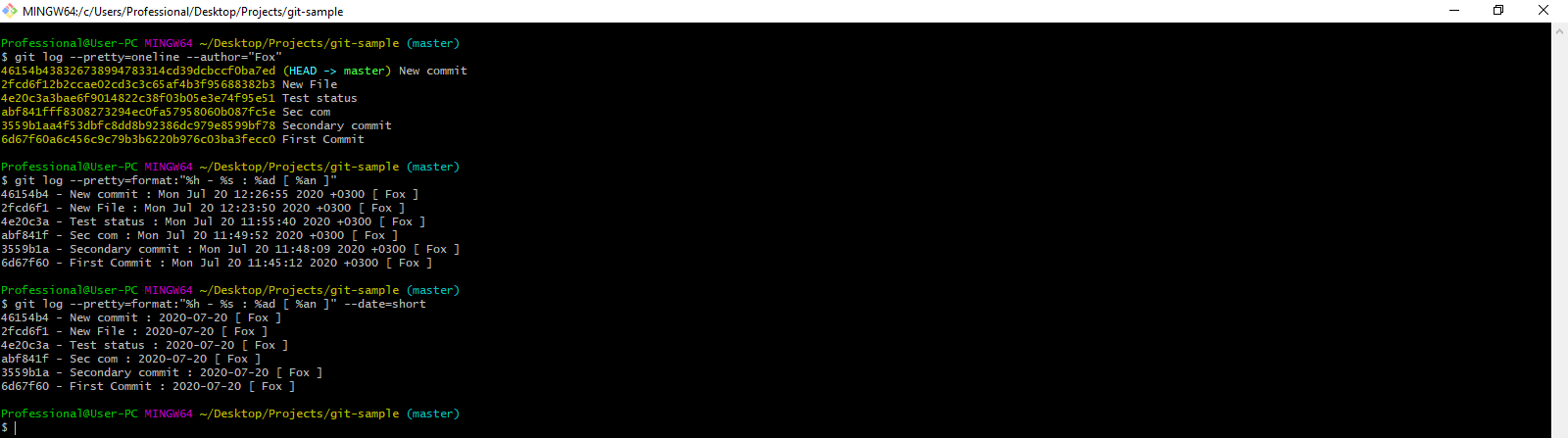
*Рис.11 – gitk*

**

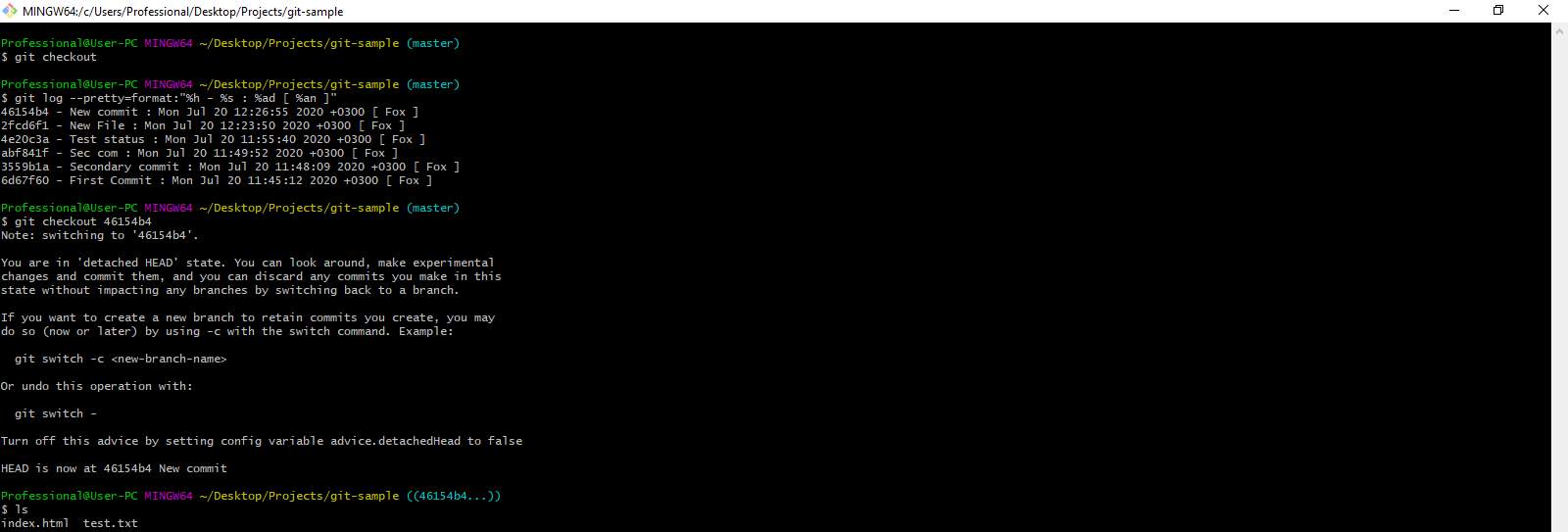
*Рис.12 – Демонстрация команды git log*

**

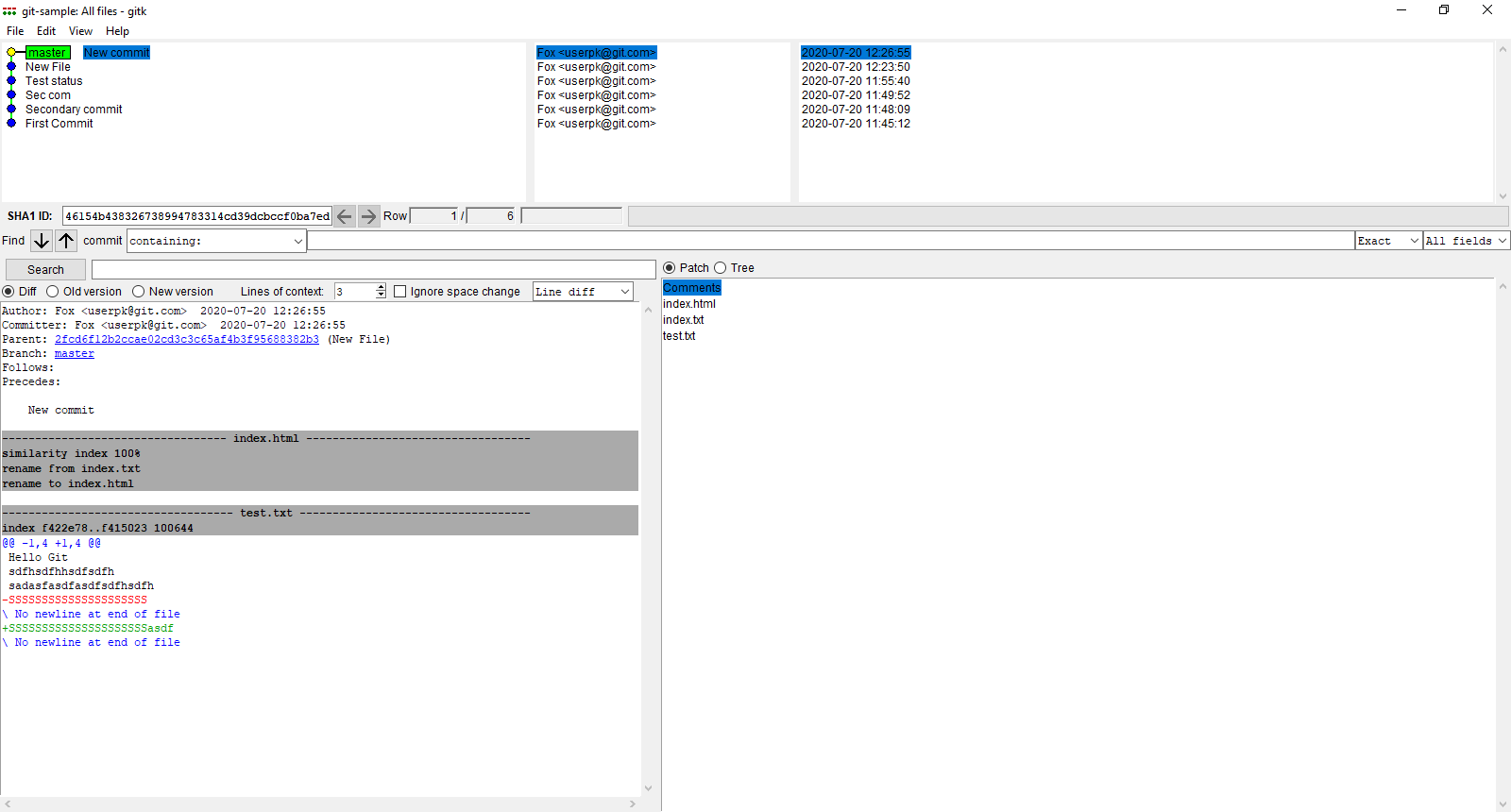
*Рис.13 – Разновидности команды git log(1)*

**

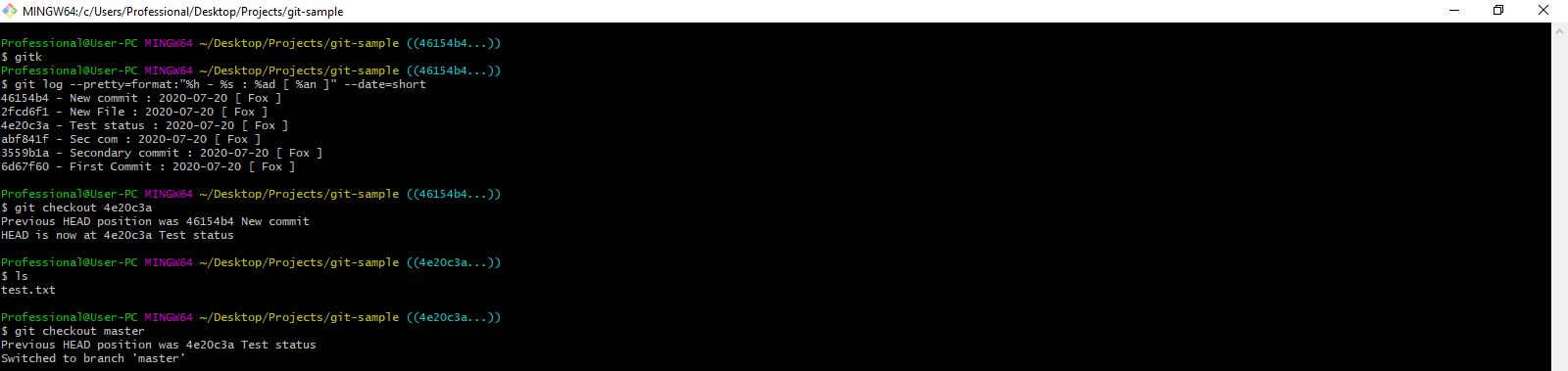
*Рис.13 – Разновидности команды git log(2)*

**

*Рис.14 – Смена ветви*

**

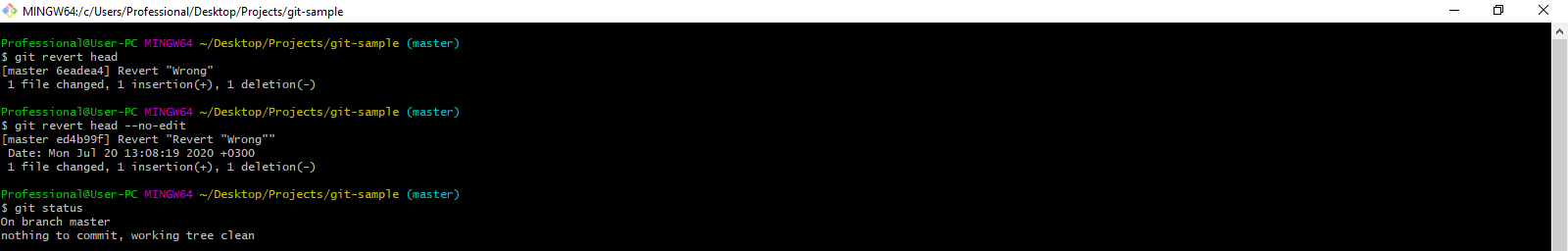
*Рис.15 – gitk*

**

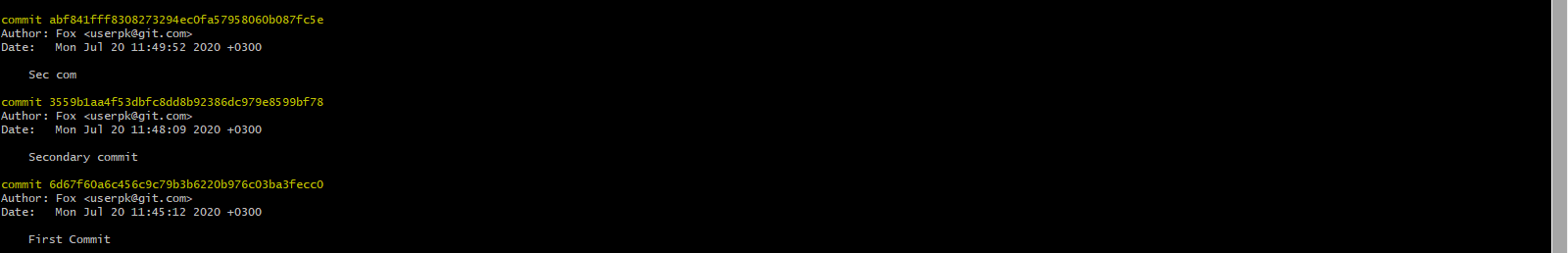
*Рис.16 – Просмотр информации о ветви*

**

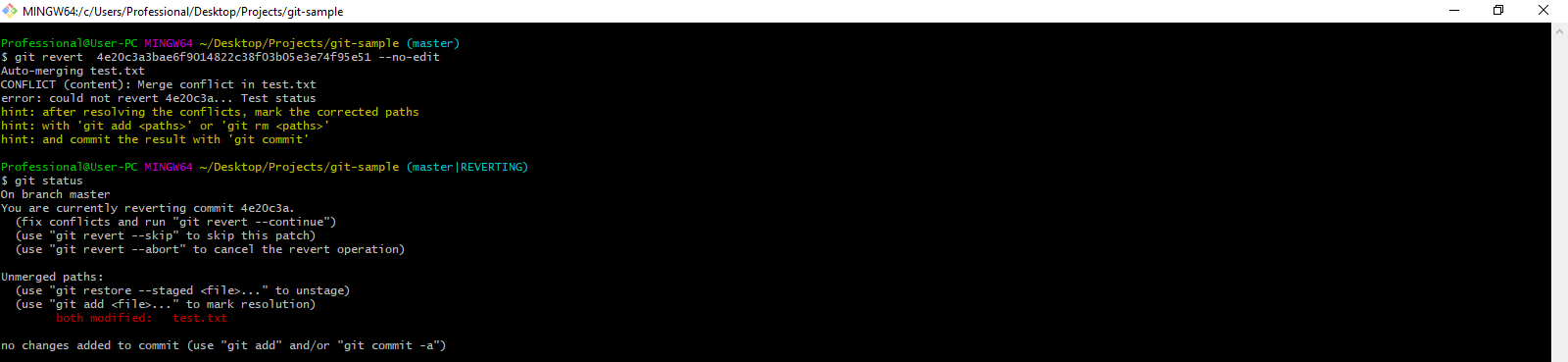
*Рис.17 – Работа с указателем HEAD*

**

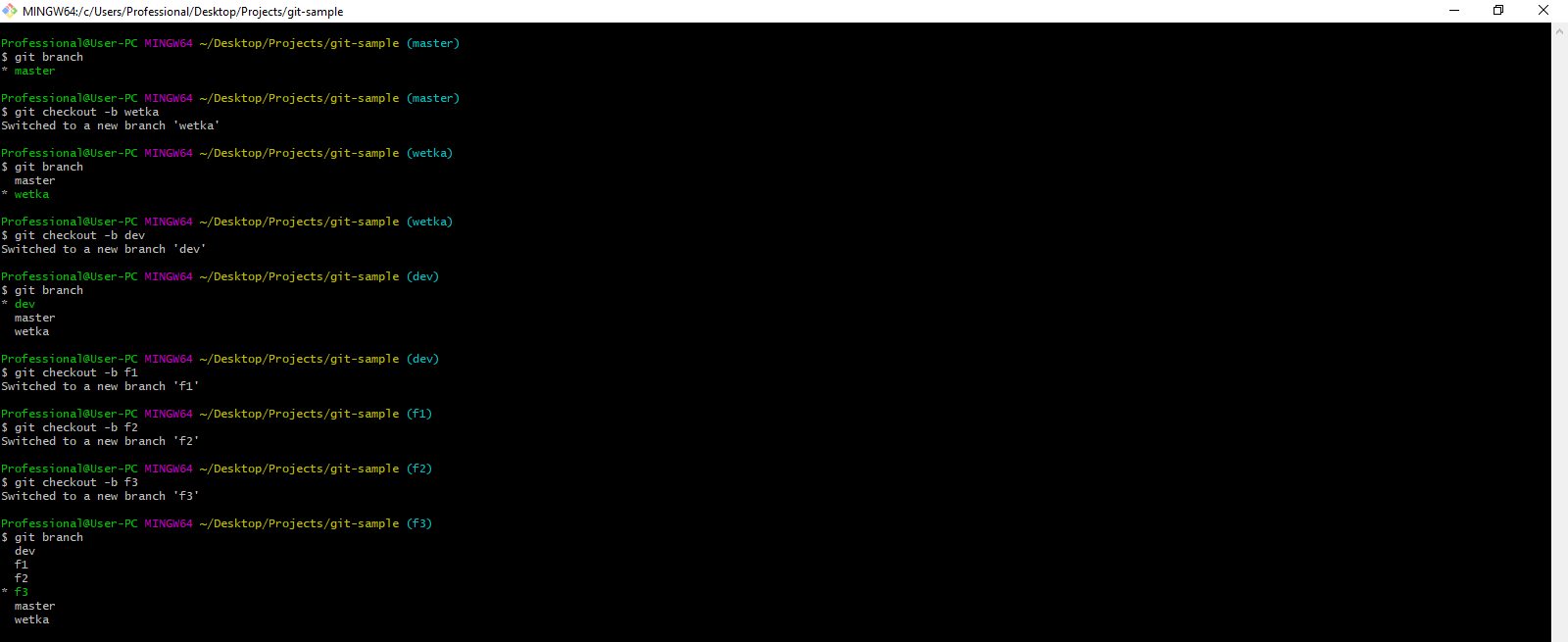
*Рис.18 – Изменение указателя HEAD*

**

*Рис.19 – git log*

**

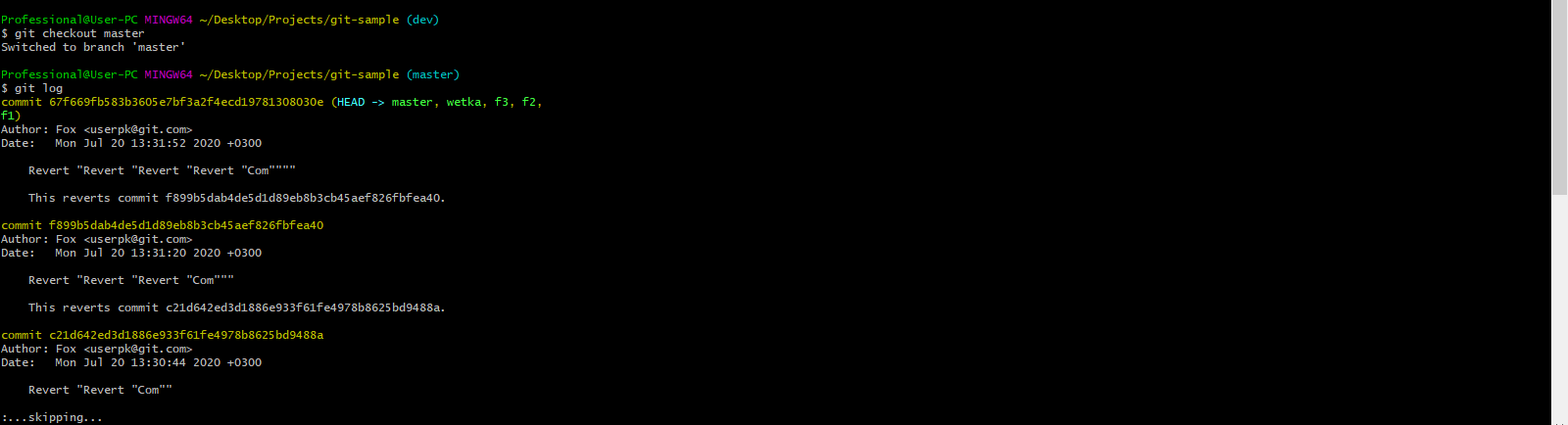
*Рис.20 – Ошибка при выполнении команды revert*

**

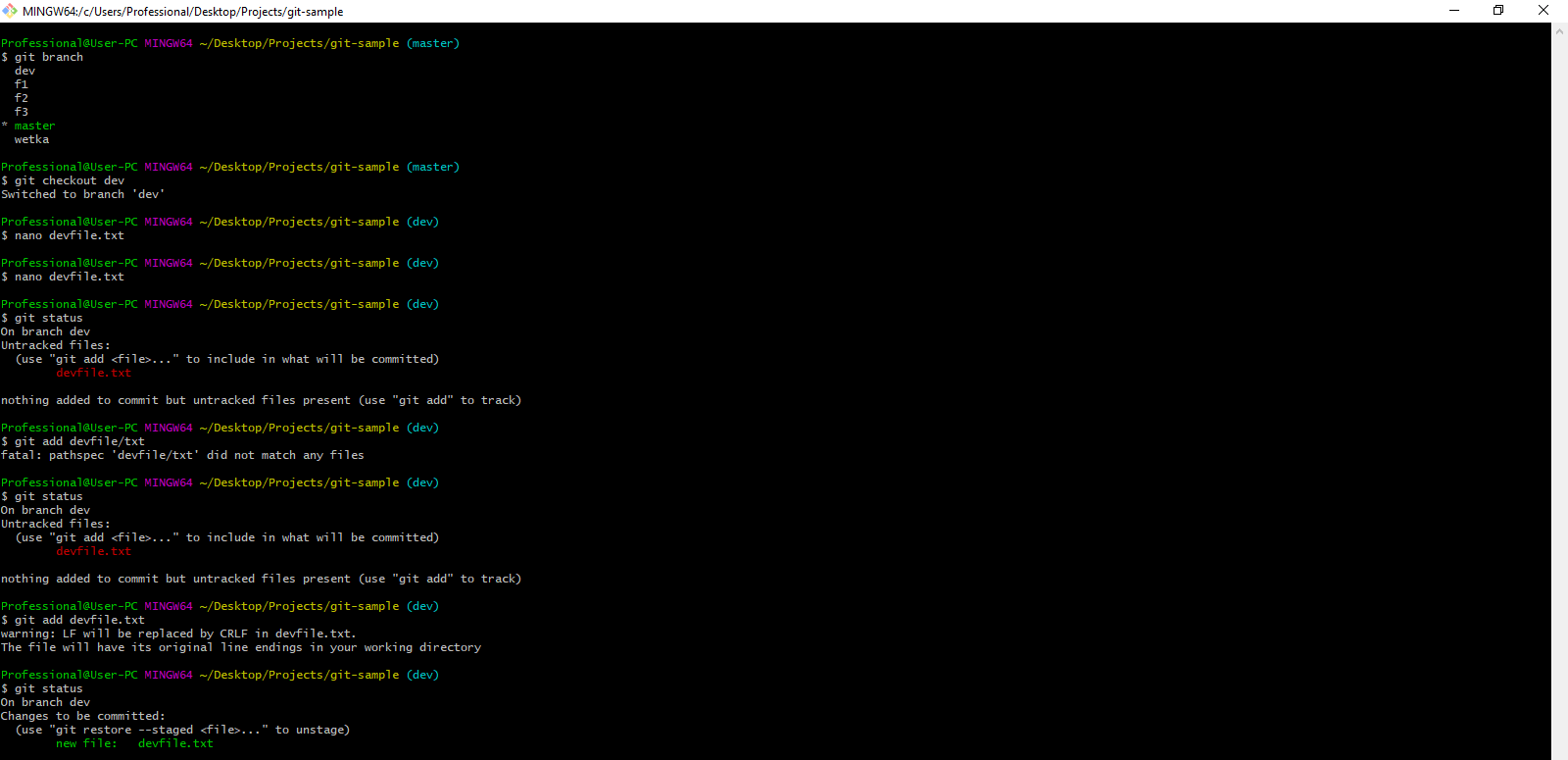
*Рис.21 – Создание ветвей*

**

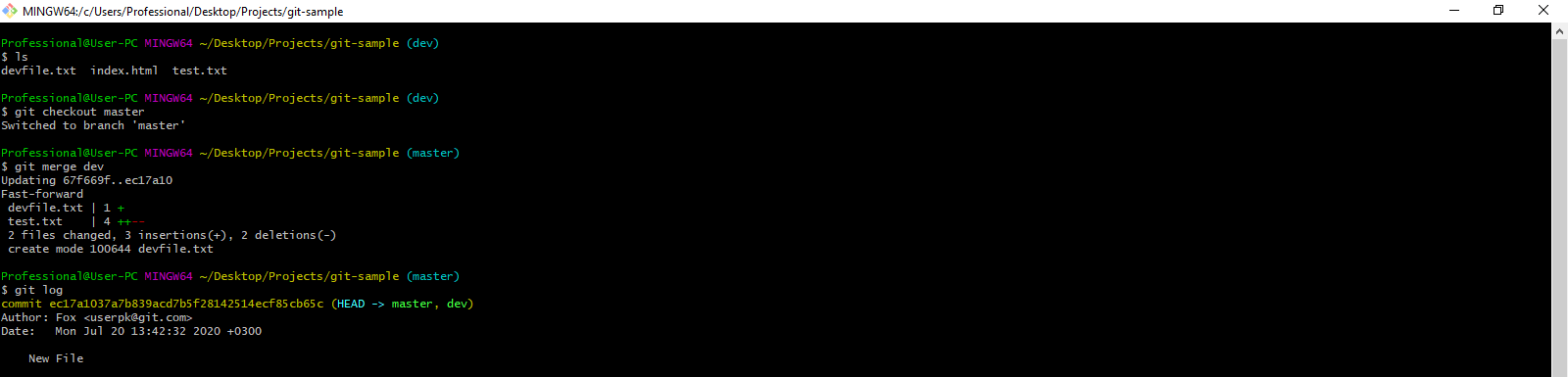
*Рис.22 – Работа с ветвями*

**

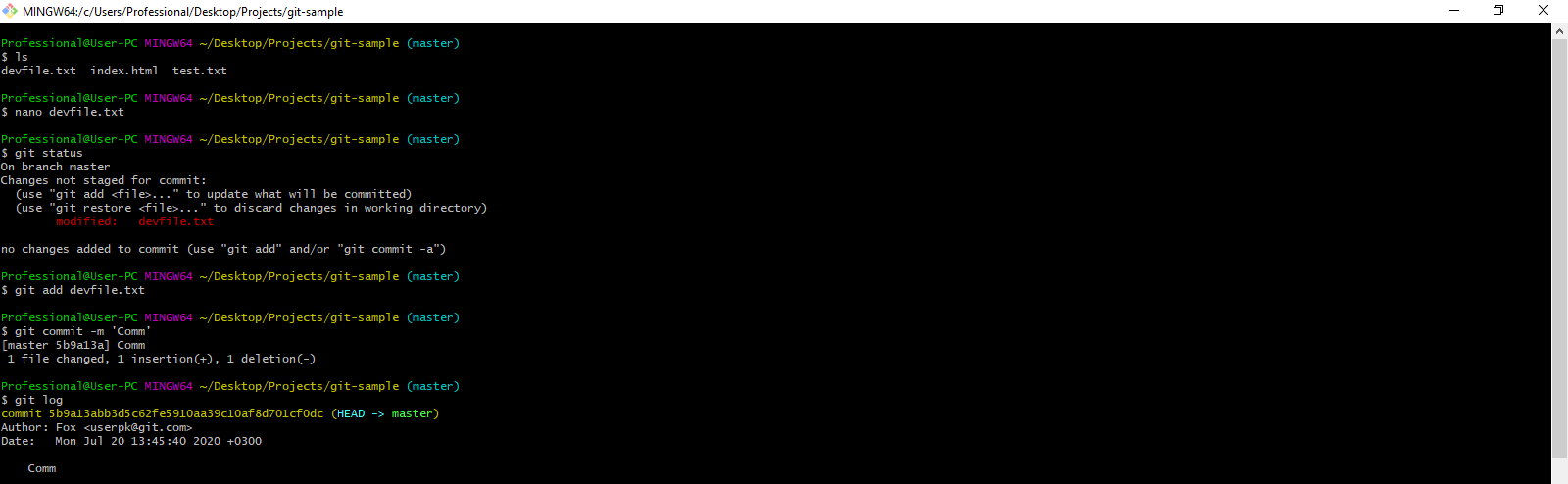
*Рис.23 – git log*

**

*Рис.24 – Создание нового текстового файла*

**

*Рис.25 – слияние ветвей*

**

*Рис.25 – Изменение файлов в соединённых ветвях*

**

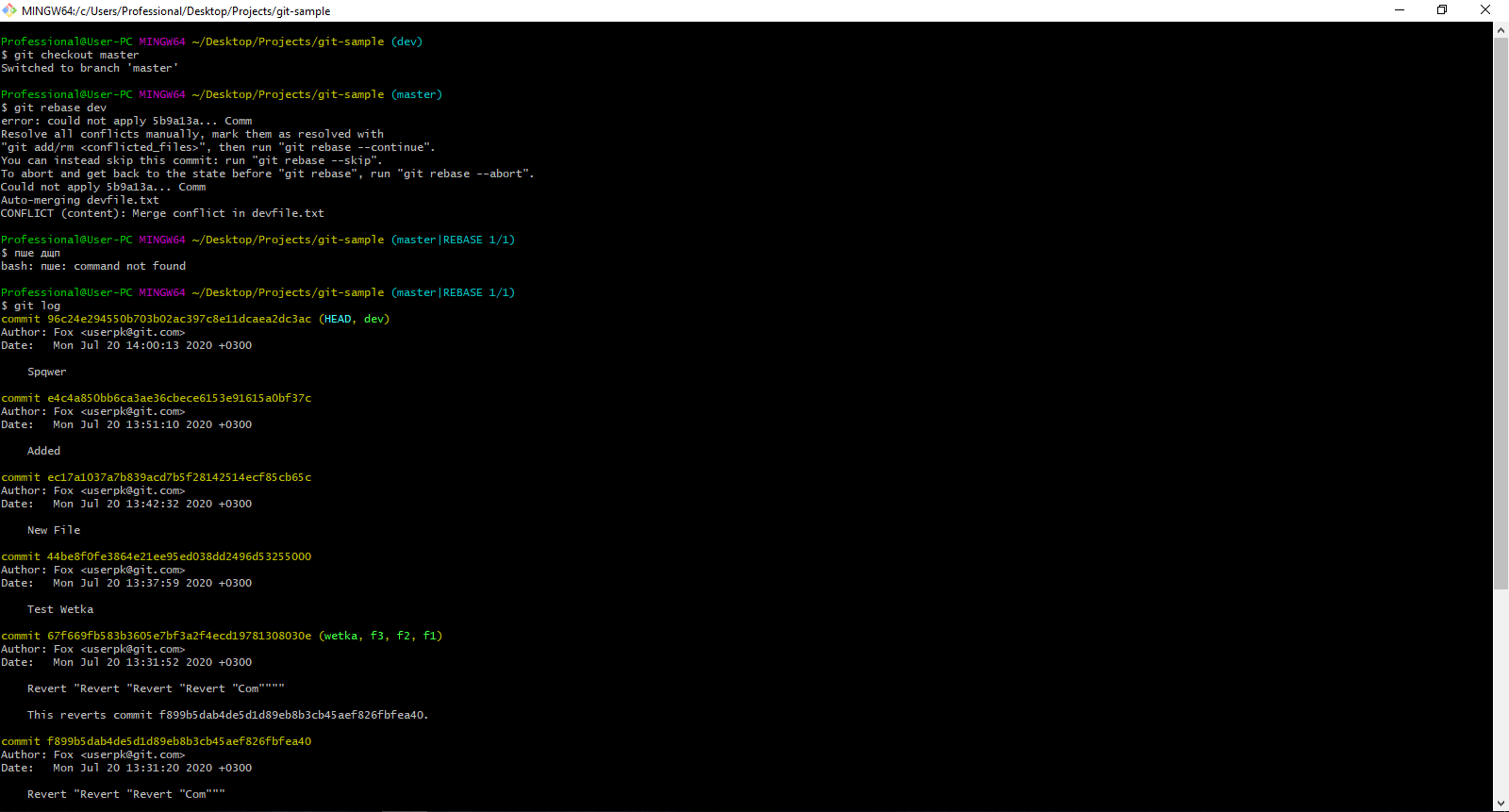
*Рис.26 – Файл devfile.txt*

**

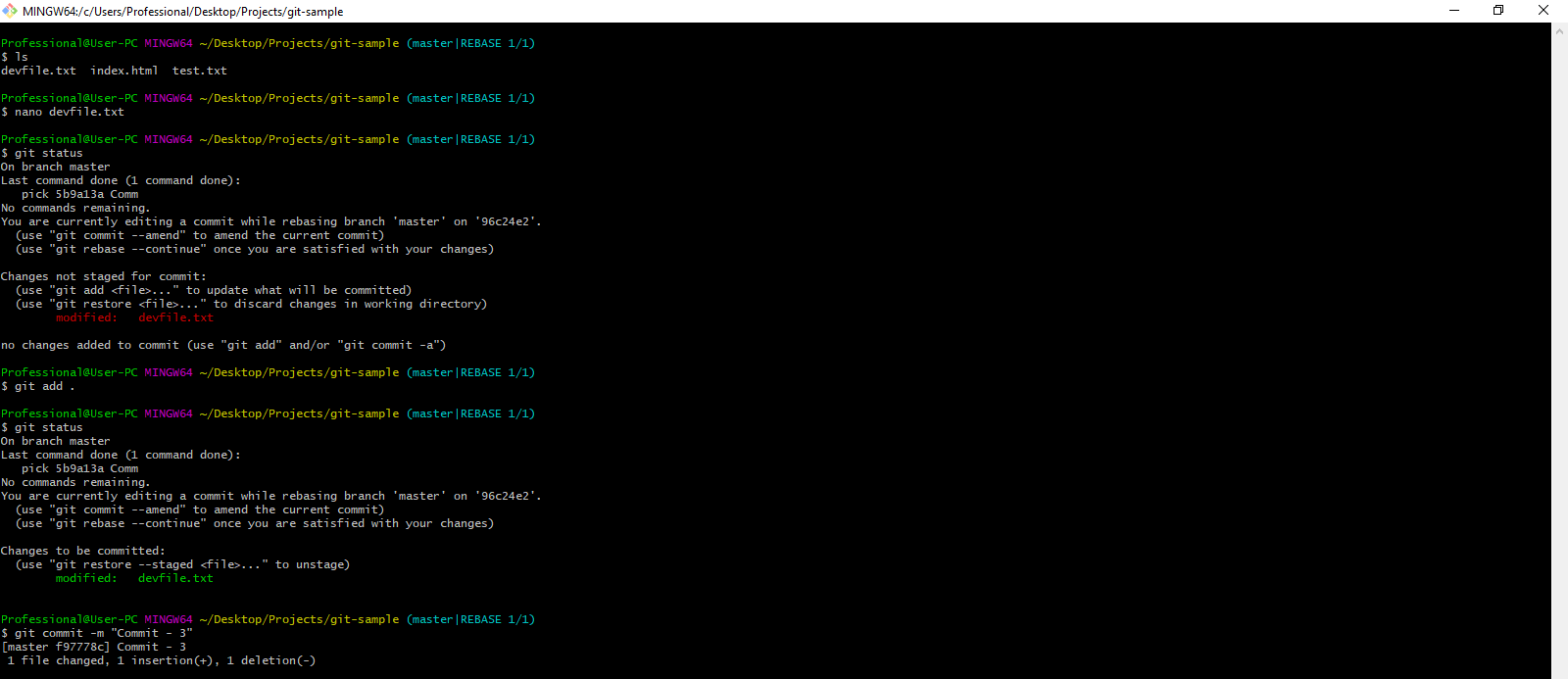
*Рис.27 – Выполнение команды merge*

**

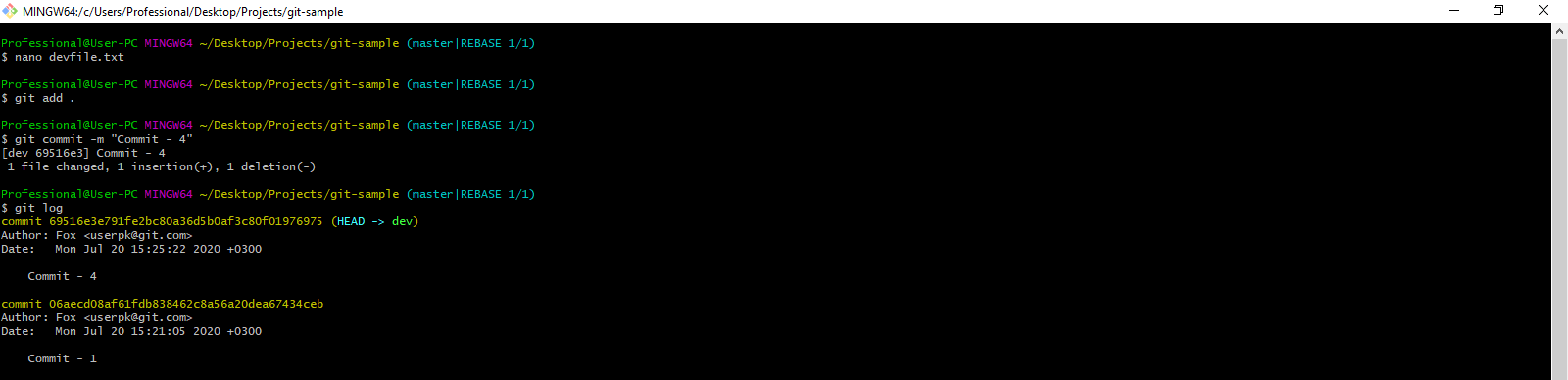
*Рис.28 – Команда git log после выполнения merge*

**

*Рис.29 – Команда rebase*

**

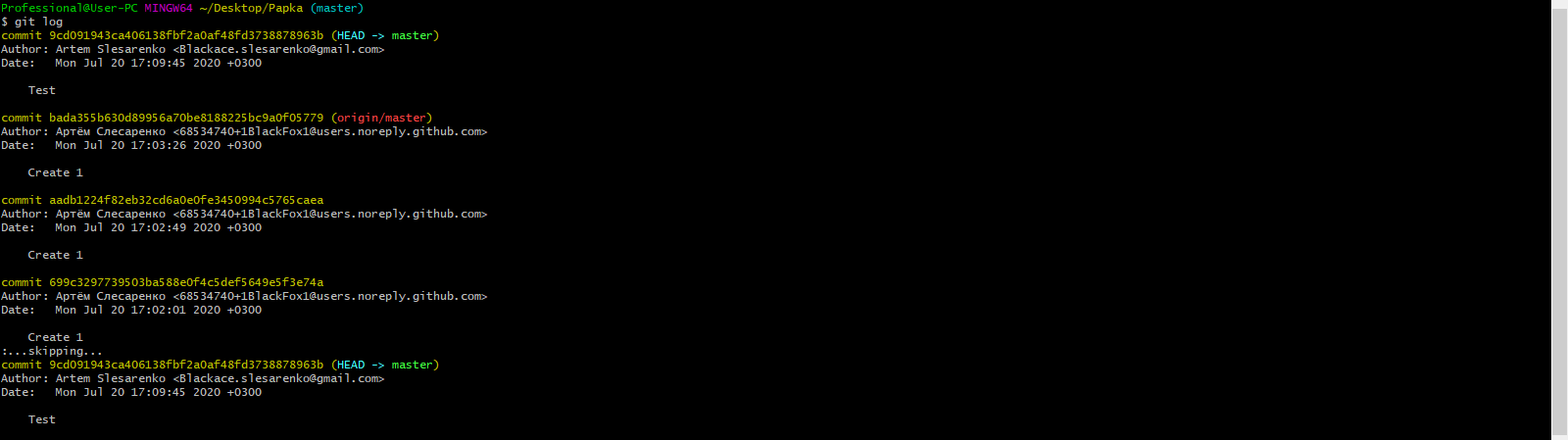
*Рис.30 – Изменение файлов в ветвях, соединённых при помощи merge(1)*

**

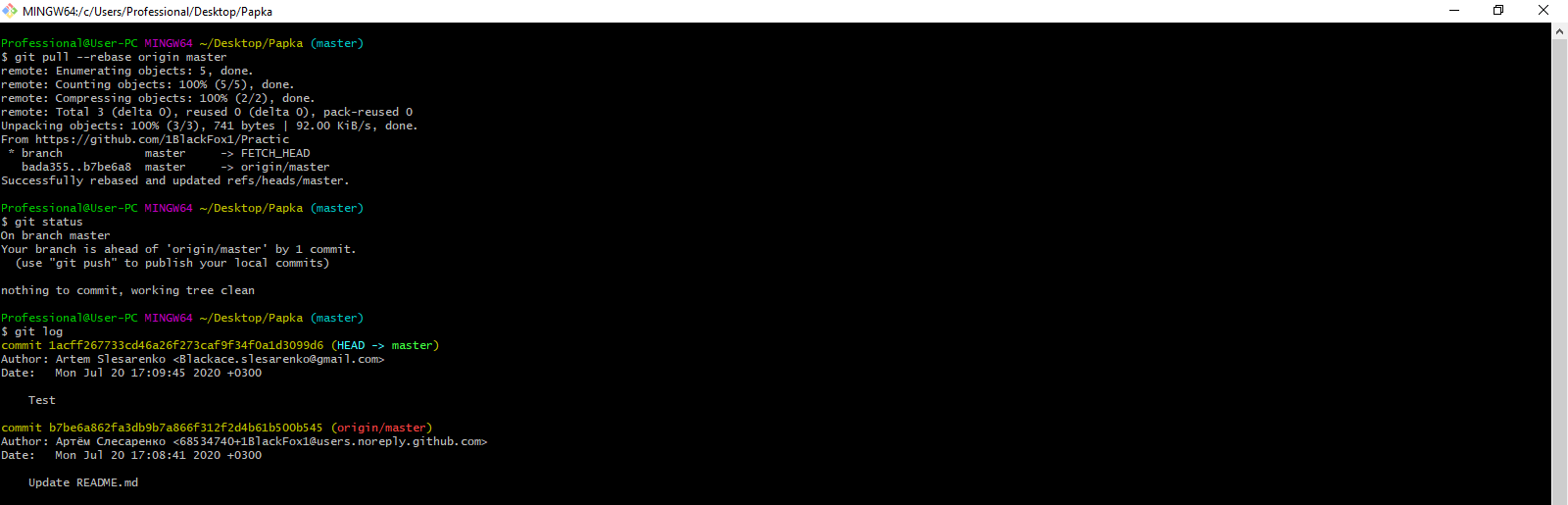
*Рис.30 – Изменение файлов в ветвях, соединённых при помощи merge(2)*

**

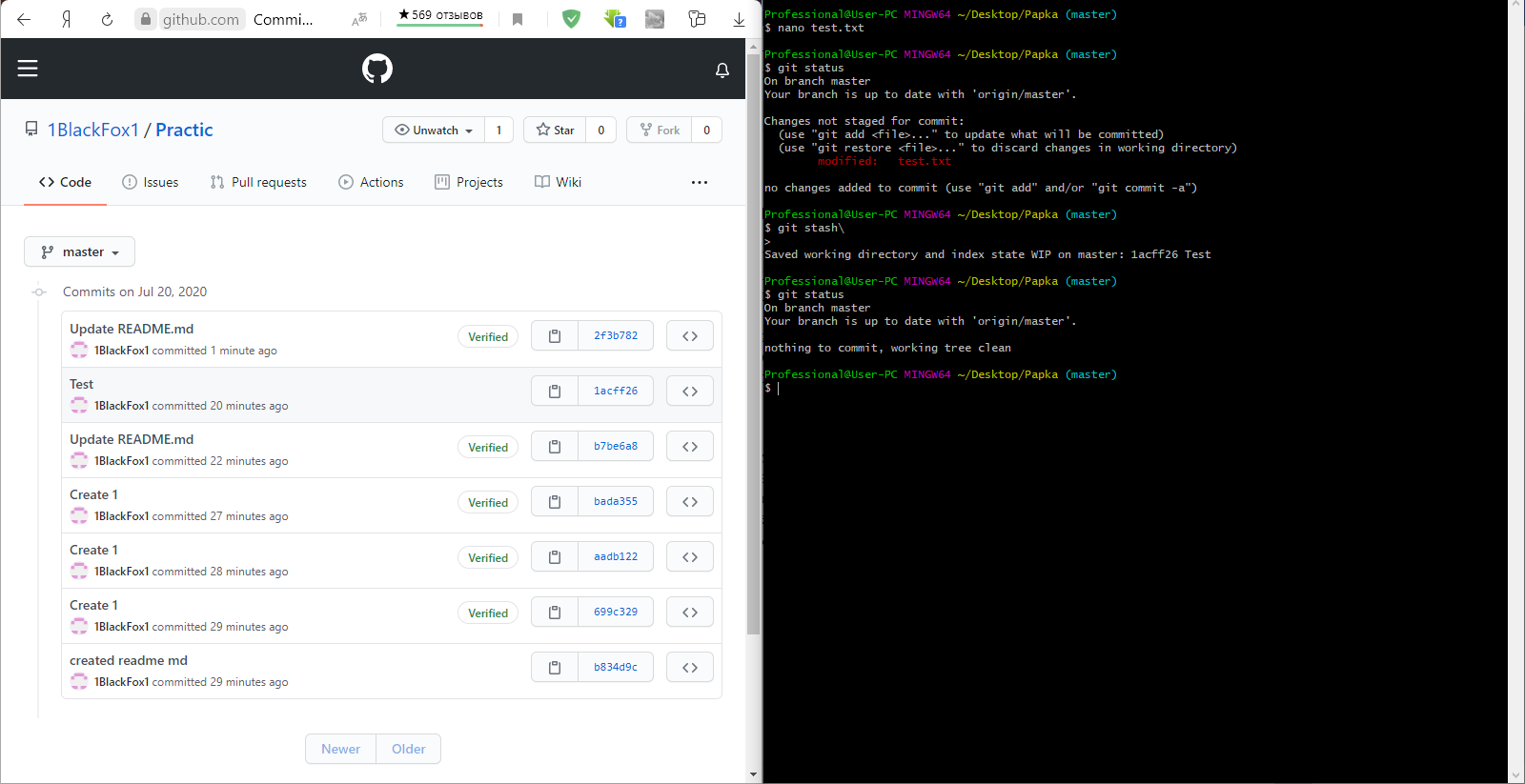
*Рис.31 – Связь с депозиторием сайта GitHub*

**

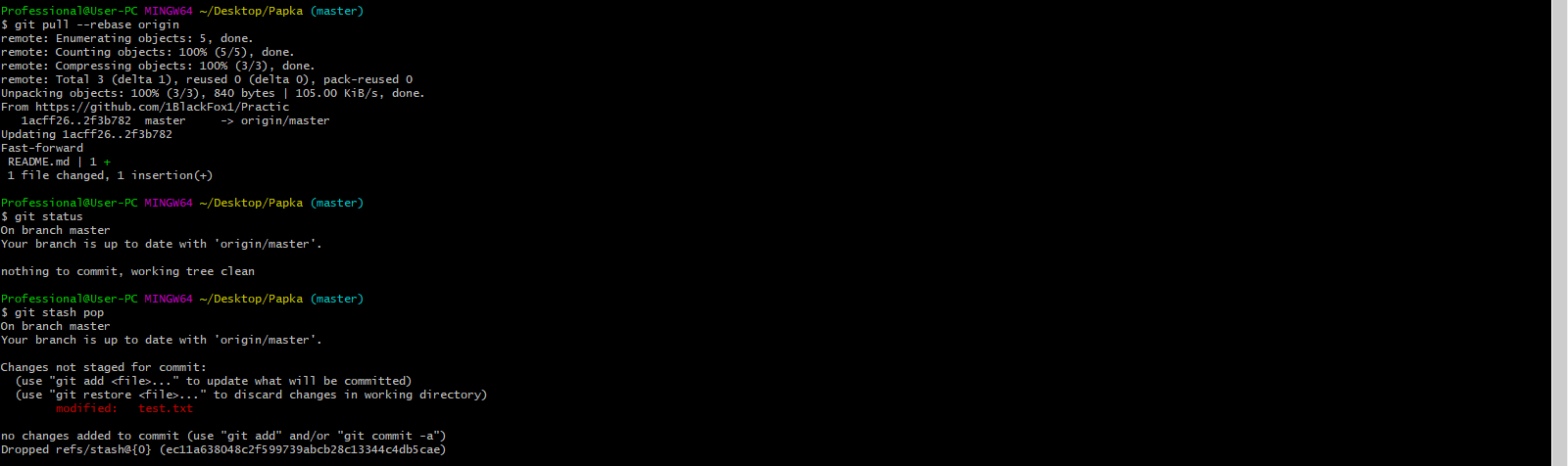
*Рис.32 – Git log*

**

*Рис.33 – Обновление онлайн депозитория*

**

*Рис.34 – Локальное и онлайн изменение разных текстовых файлов*

**

*Рис.35 – Обновление депозитория после изменения локальных файлов(1)*

**Вывод:** научился использовать набор утилит Git; изучил консольные команды с помощью приложения Git Bash; создал онлайн депозиторий на сайте GitHub.