Міністерство освіти і науки України

Харківський національний університет імені В.Н. Каразіна

Факультет комп’ютерних наук

**ЛАБОРАТОРНА РОБОТА №1**

з дисципліни

«Математичні методи та технології тестування та верифікації програмного забезпечення»

Тема: Вивчення системи контролю версій на прикладі Git.

Виконав студент 2 курсу

групи КС-21

Безрук Юрій Русланович

Перевірив: доцент Нарєжній О.П.

Харків – 2020

Целью данной работы является получение базовых знаний о системе Git и чем он отличается от централизованных систем контроля версий, а также получение рабочей версии Git на компьютере, настроенной и персонализированной.

ХОД РАБОТЫ

1. Установка Git на своем компьютере.

Система Git была скачана с ресурса <https://git-scm.com/download/win> и установлена на компьютер. Далее будет проводиться предварительная настройка системы. Результат запуска Git Bash:

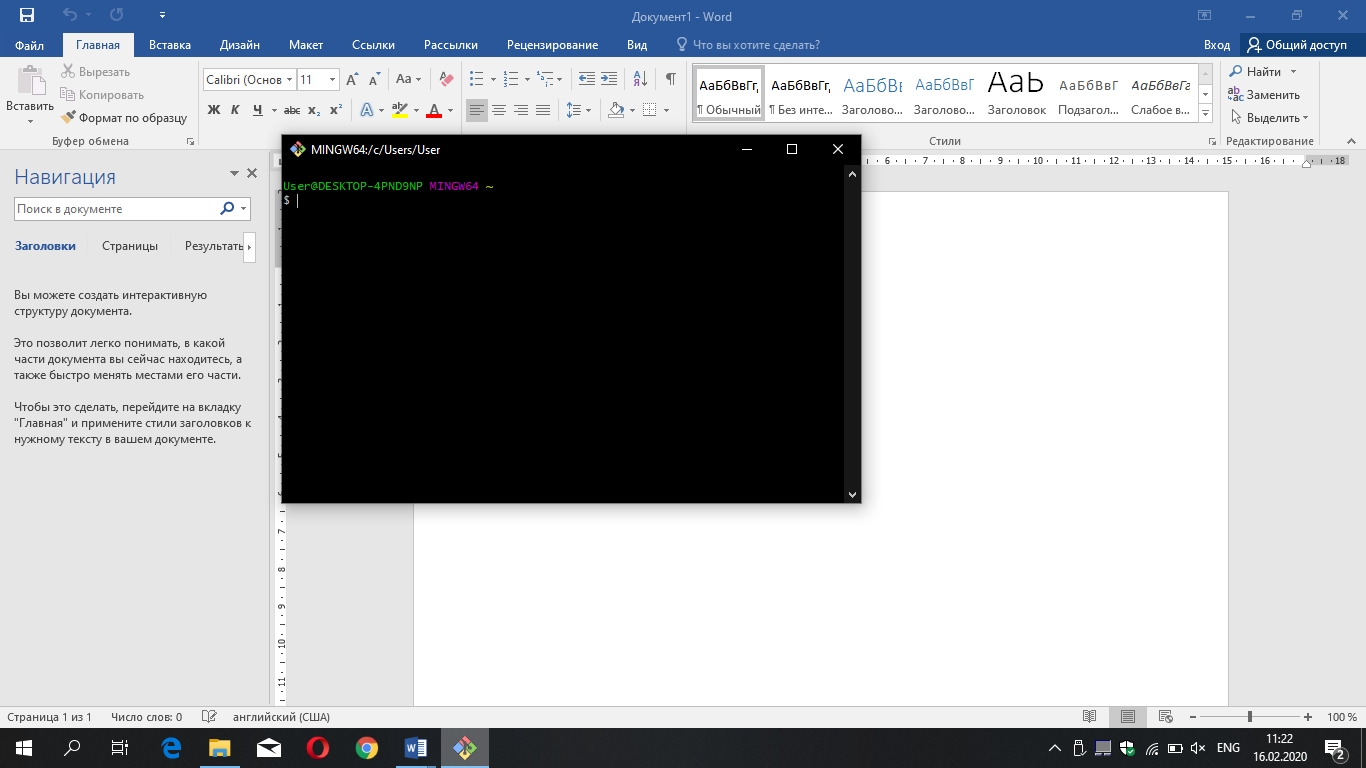
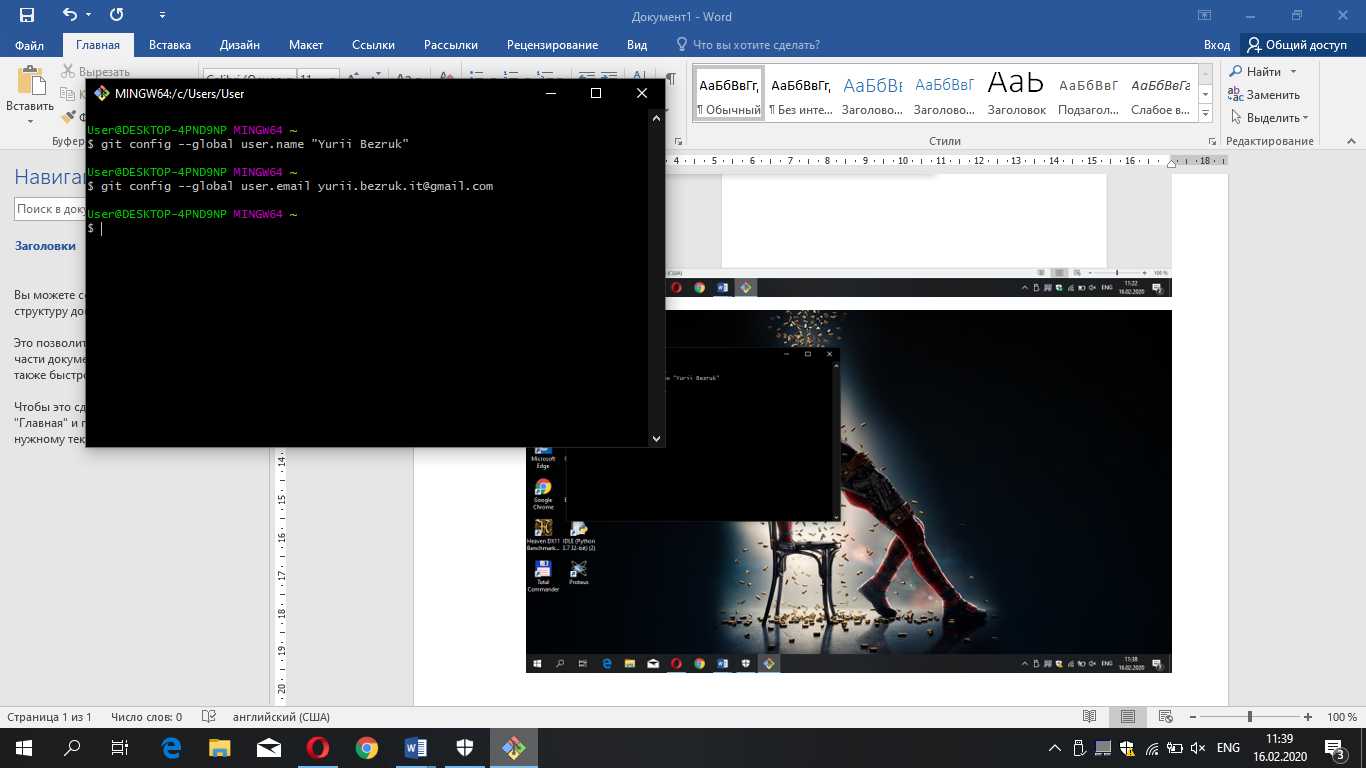
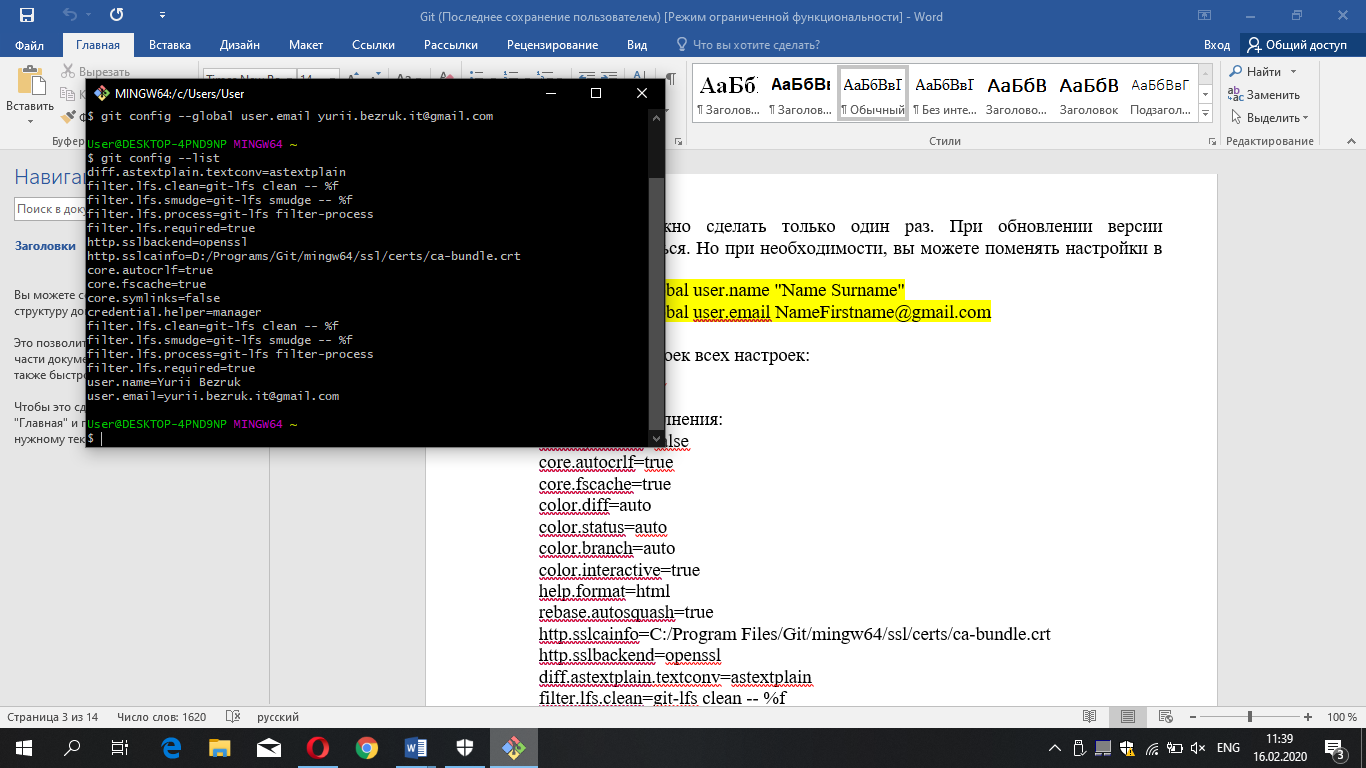


Рисунок 1 - Интерфейс Git Bash

1. Первоначальная настройка Git

К первоначальной настройке относится ввод пользовательских данных – имени и електронной почты, а также проверка установленных кофигураций Git.

Рисунок 2 - Предварительная настройка Git



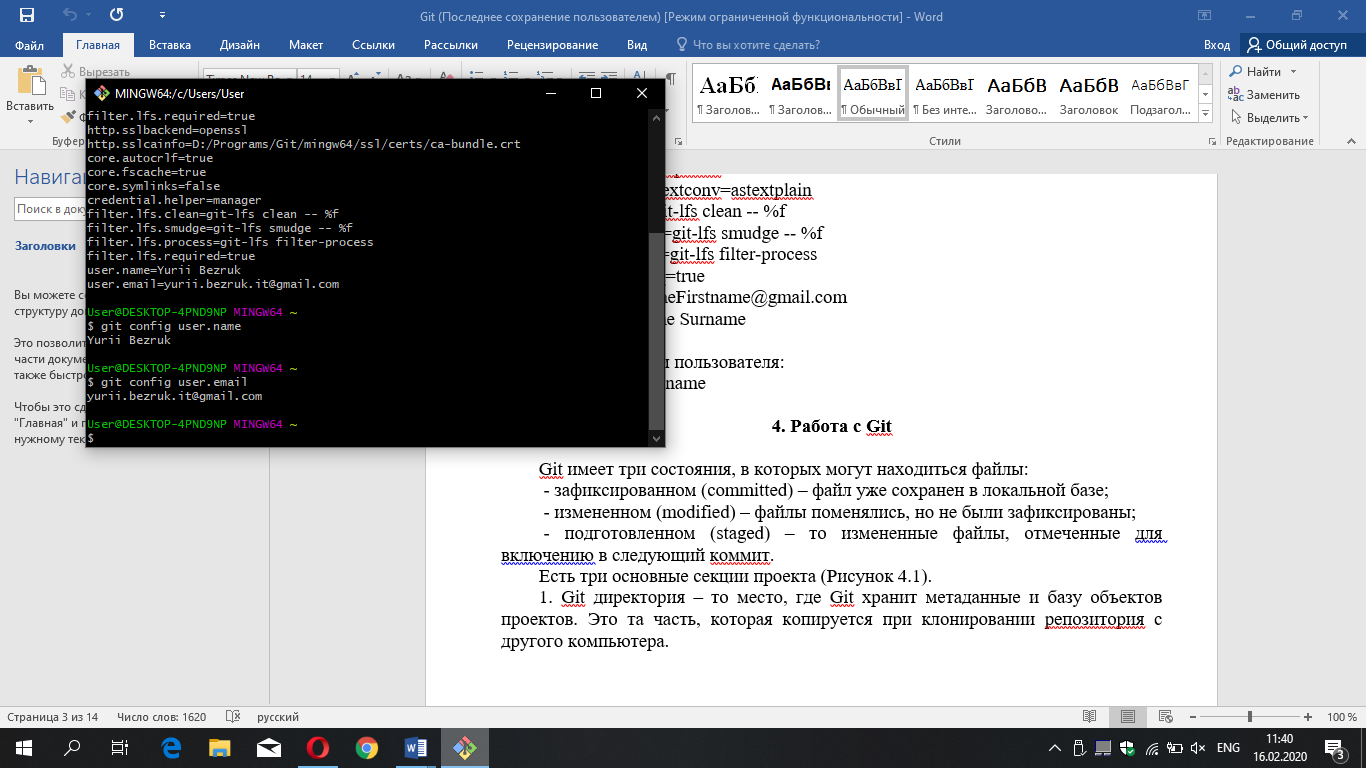


Рисунок 3 - Проверка данных

После чего производится запуск Git в папке проекта для тестирования. Выполняется команда $git init, создающая репозиторий из папки. Проверяется статус репозитория, создаётся файл для тестирования – file.txt. Файл индексируется при помощи командым $git add, что значит, что все изменения в нем отныне отслеживаются Git.

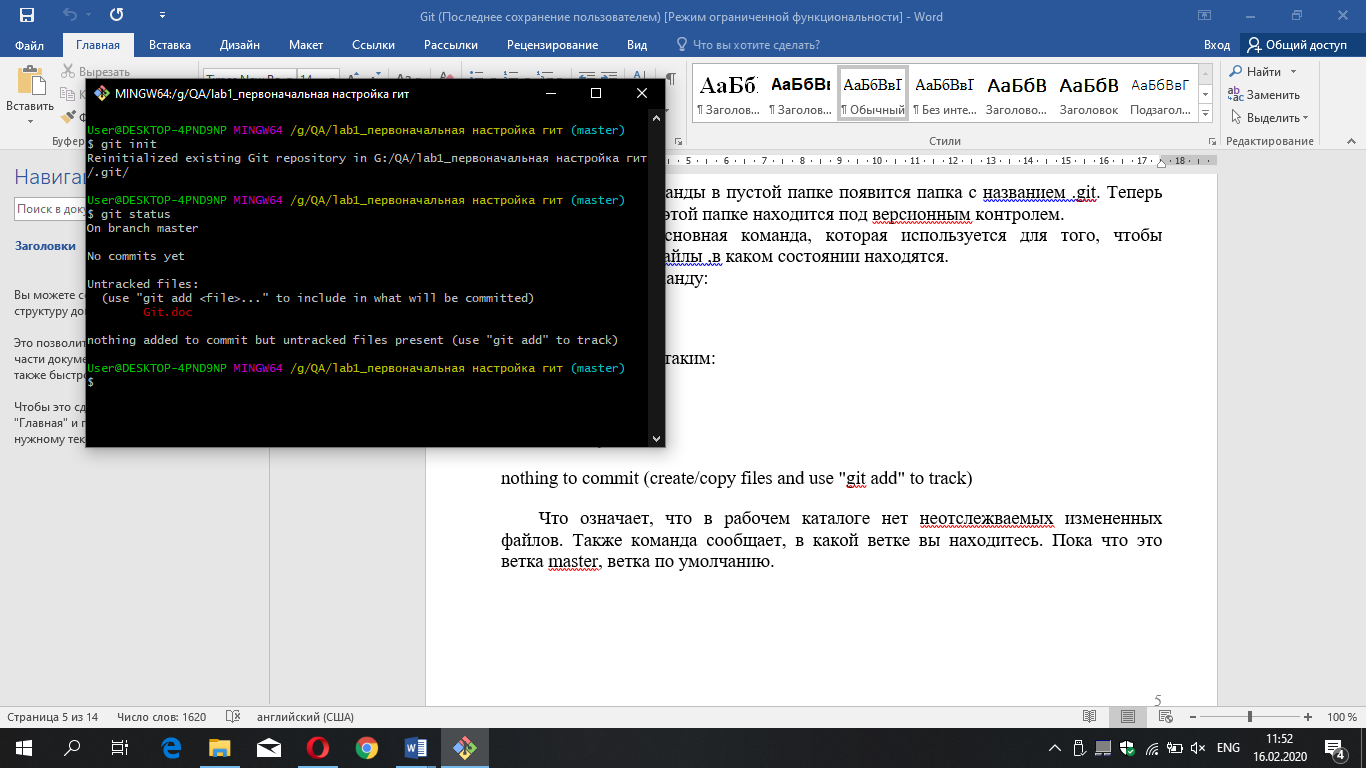


Рисунок 4 - Инициализация репозитория

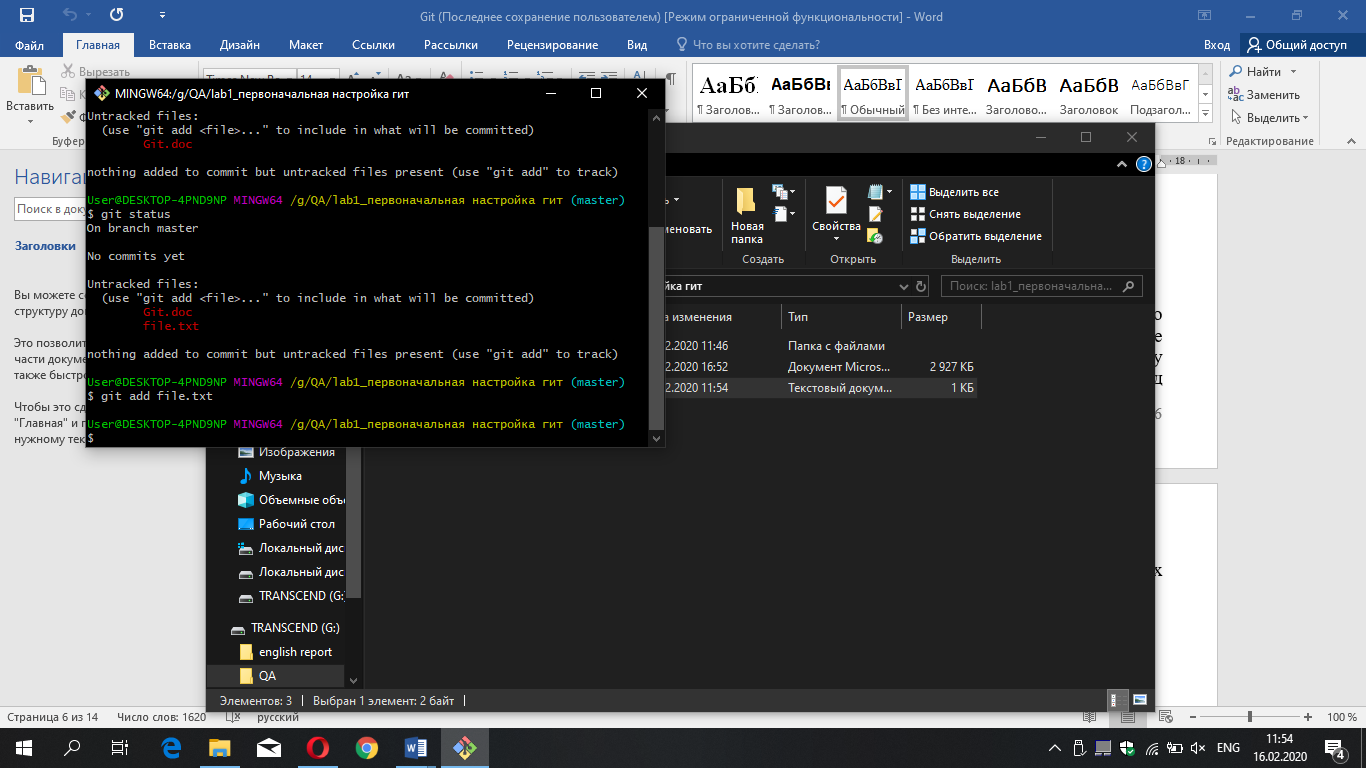


Рисунок 5 - Индексация файла

После успешной индексации статус файла изменяется на «отслеживаемый».

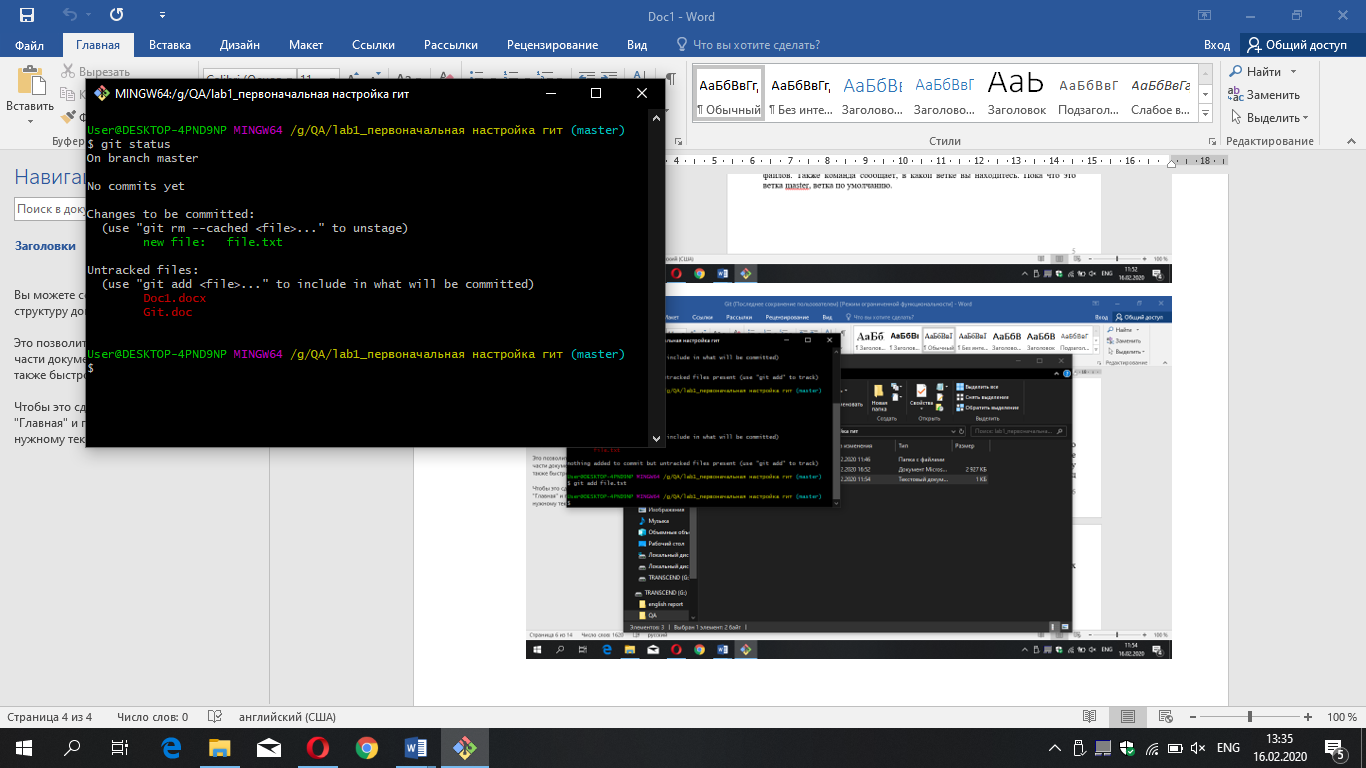


Рисунок 6 - Статус файла

1. Создание коммитов

После создания файла необходимо протестировать реакцию системы на изменение файла. Для этого в текстовый файл были записаны несколько символов. При повторном вводе $git status состояние файла изменилось на «изменён».

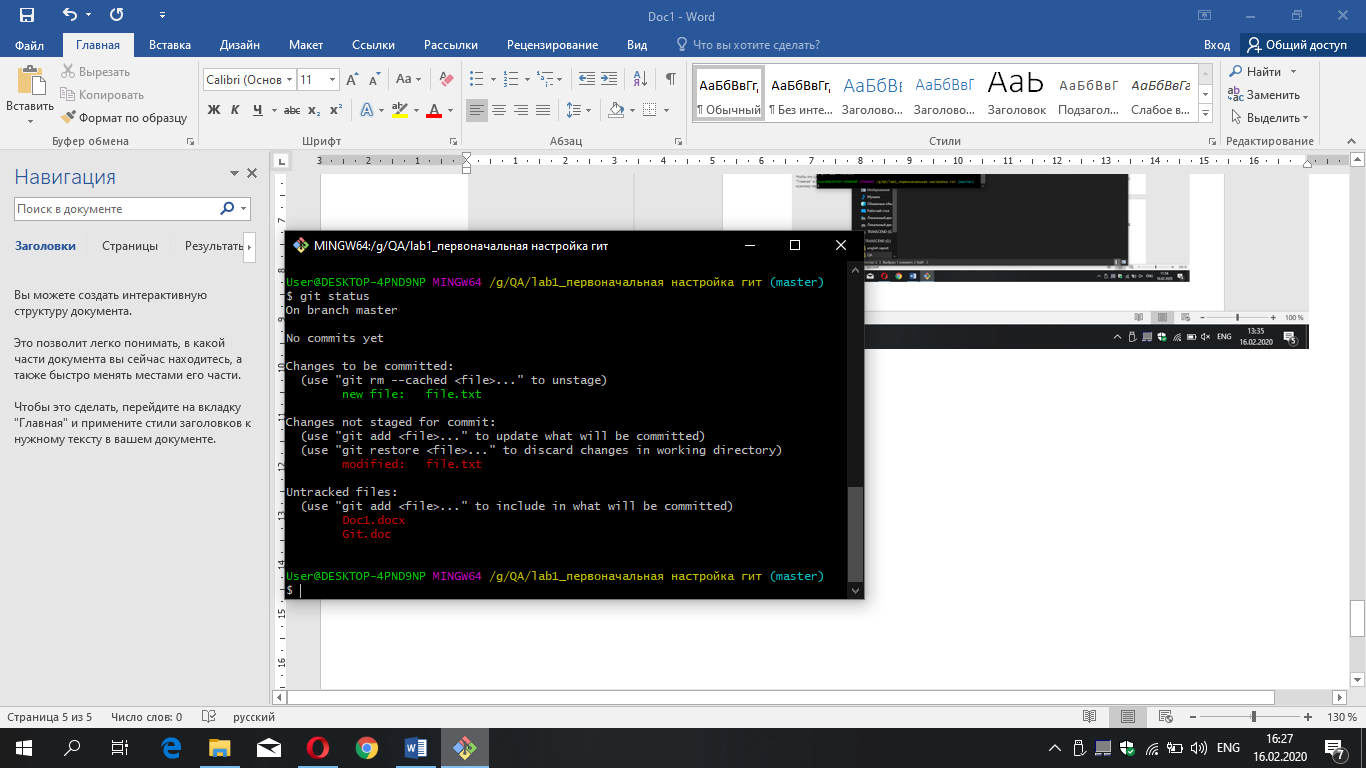


Рисунок 7 - Отслеживание изменения

Что бы зафиксировать изменения используется коммит, сохраняющий данные об изменениях в файле. После коммита состояние файла изменяется снова и система указывает что из отслеживаемых файлов изменённых нет.

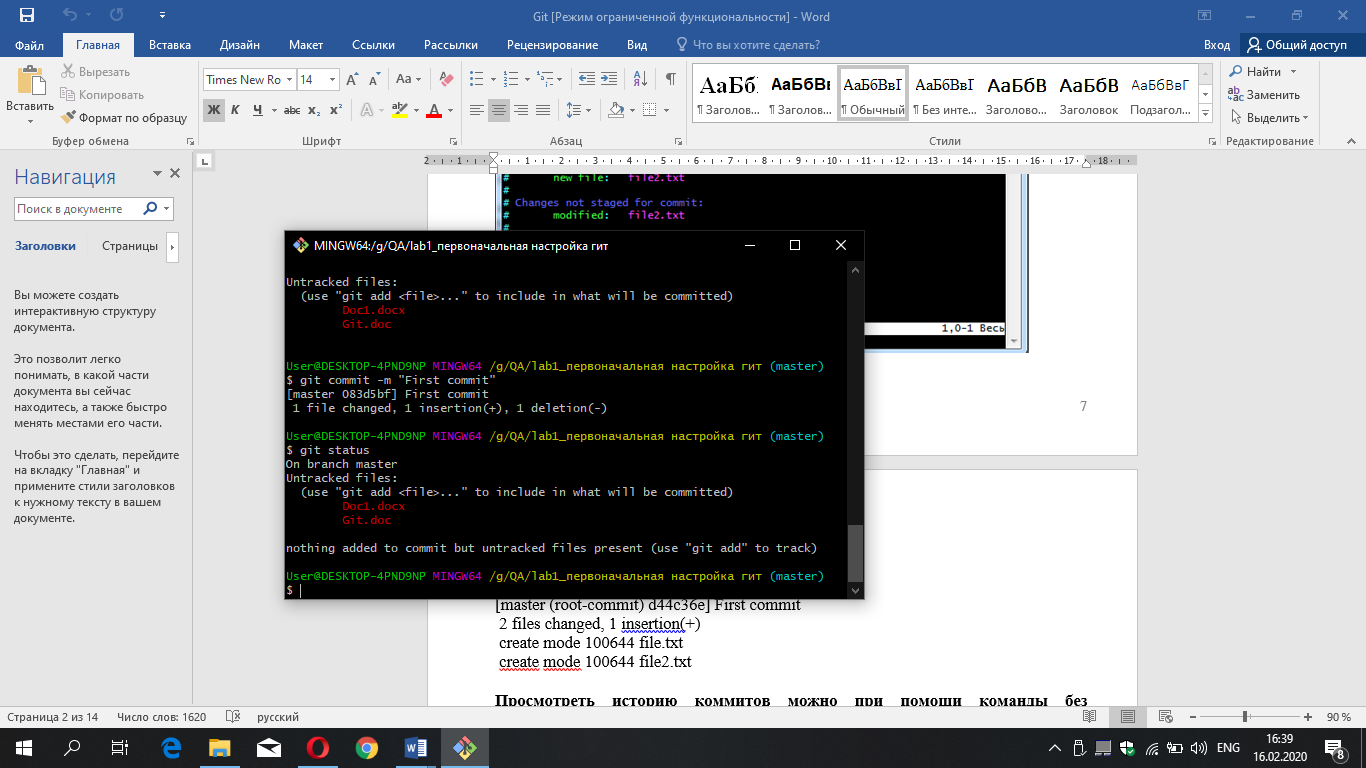


Рисунок 8 – Коммит

1. Просмотр истории изменений в файлах

Что бы узнать историю всех коммитов, необходимо ввести команду $git log. Она выдает основную информацию о последних коммитах.

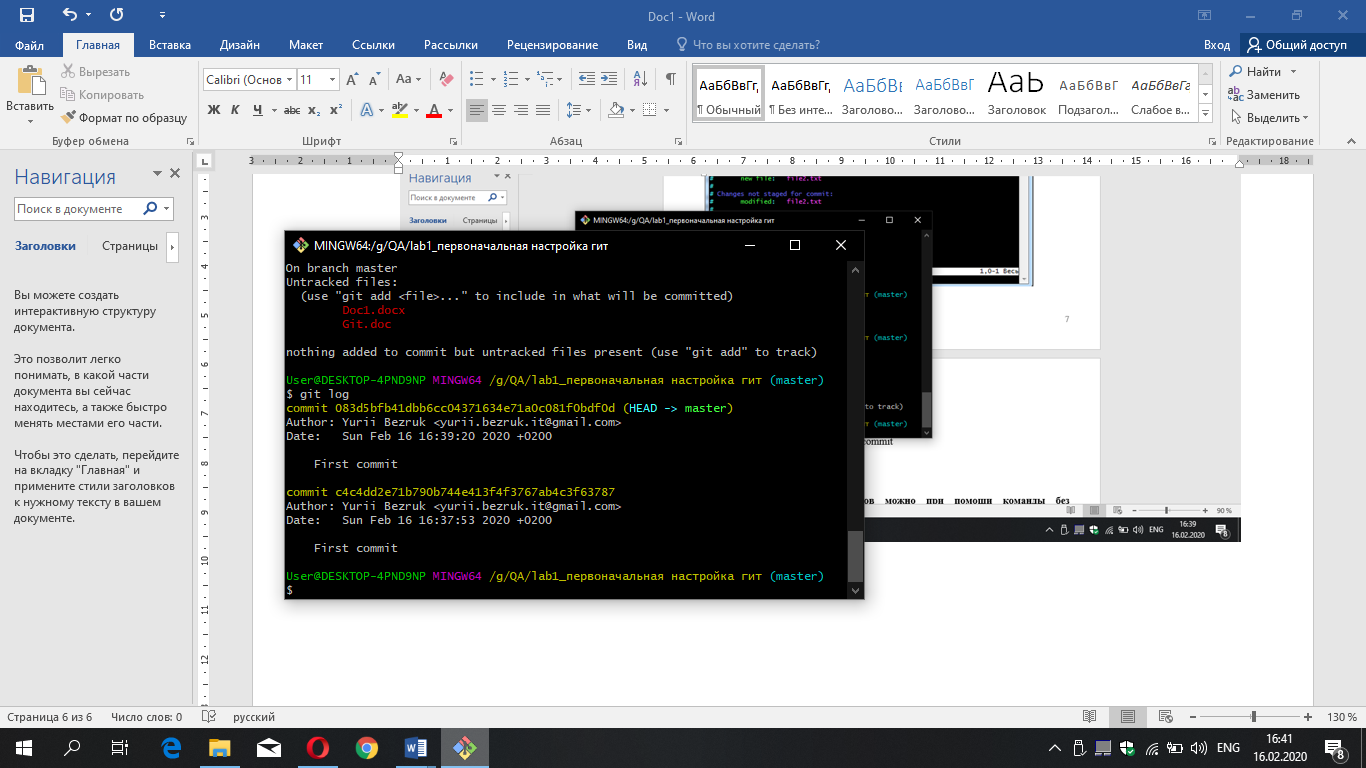


Рисунок 9 - История коммитов

1. Создание веток

Для удобной реализации резервных копий в Git используется система ветвления. Для создания новой ветки используется команда $git branch, а для переключения – $git checkout. Комбинируя ветвление и коммиты можно удобно совершать версионный контроль над проектом. Создаётся новая ветка и выплняется переход на неё, после чего файл индексируется уже в новой ветке, и выполняется коммит.

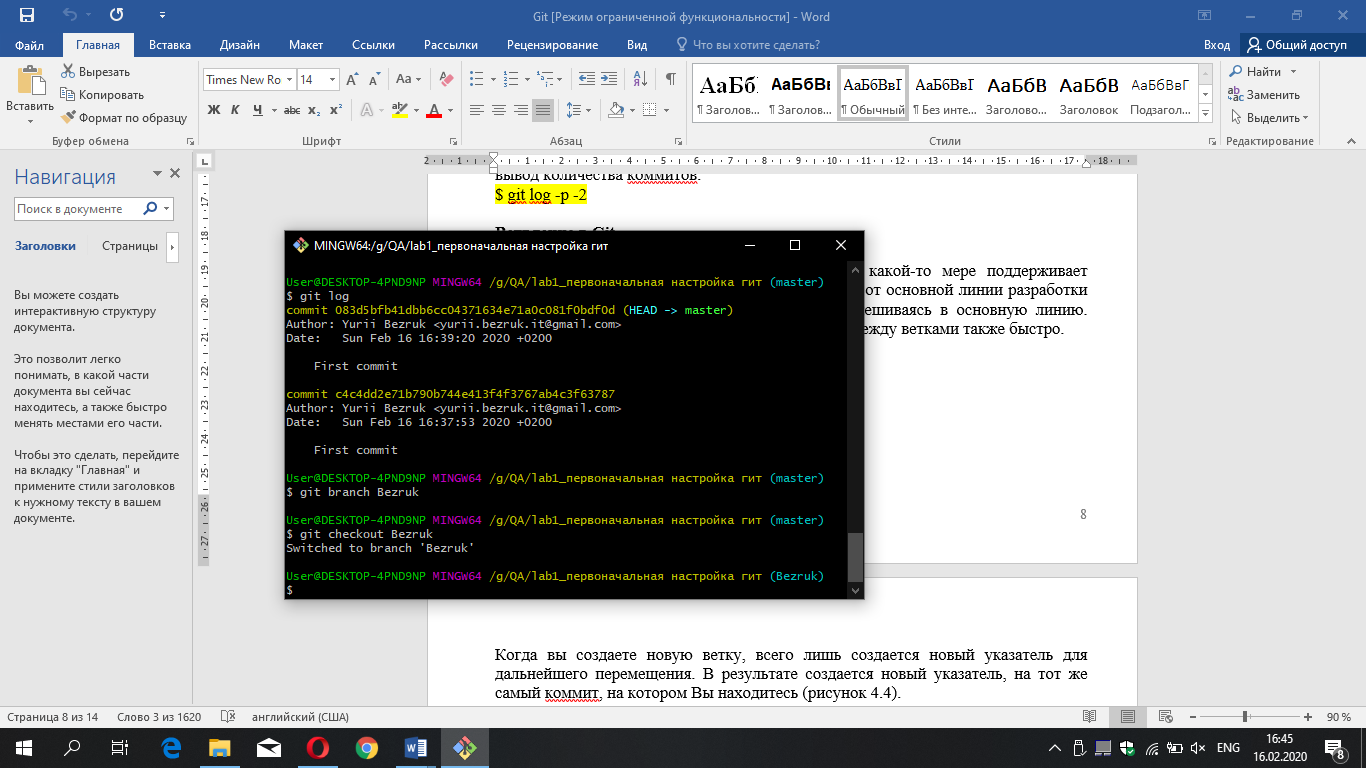
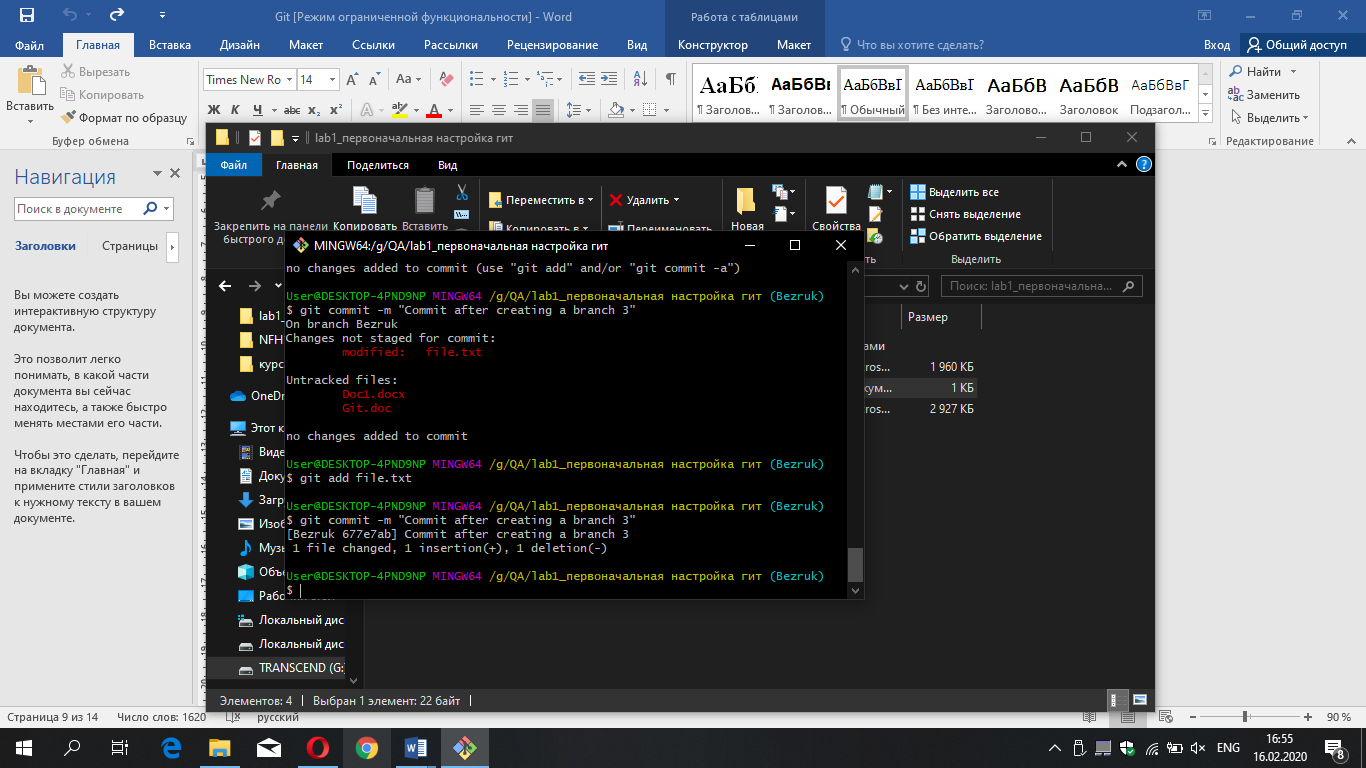


Рисунок 10 - переход на новую ветку

Теперь на ветках master и Bezruk хранятся разные версии файла, и переход на мастер-ветку будет означать откат изменений. Аналогичным образом изменения в файле записываются с помощью еще нескольких коммитов.

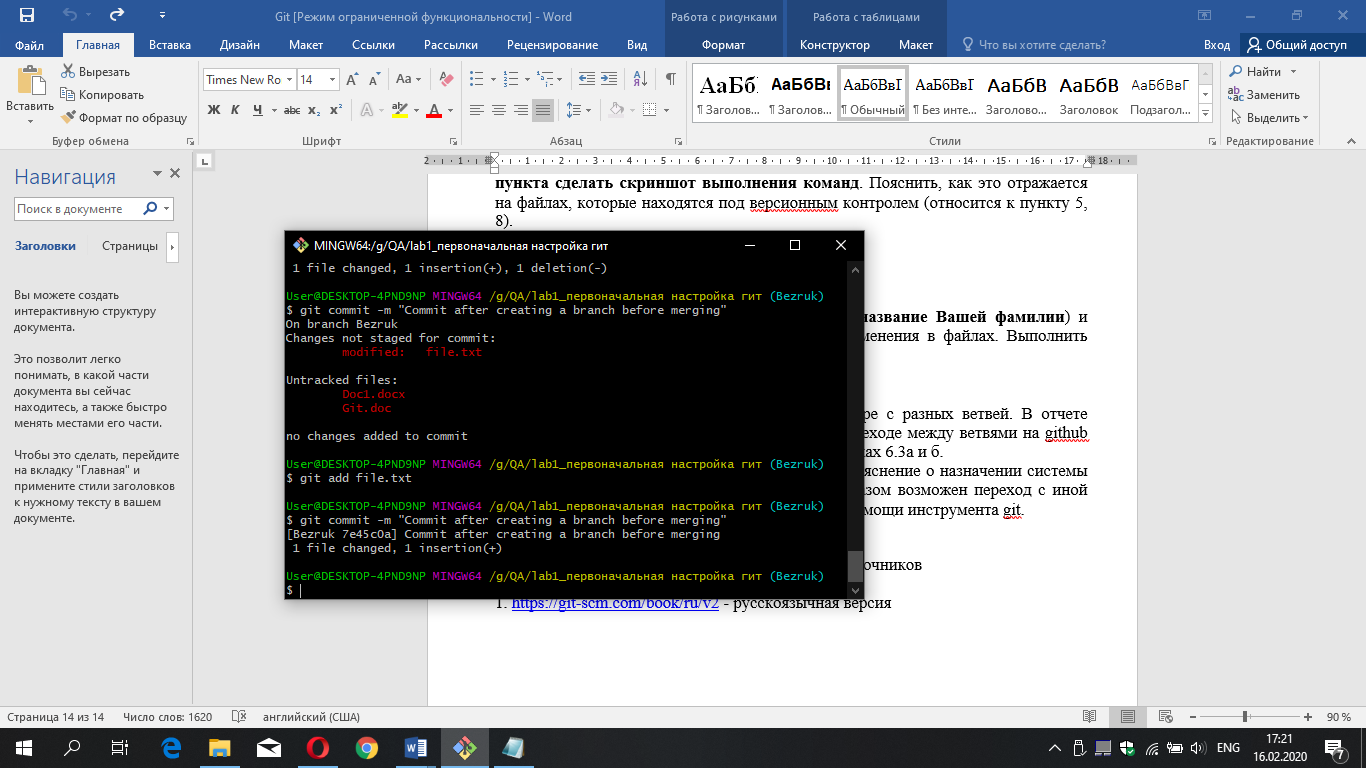


Рисунок 11 - модификация файла на новой ветке

1. Слияние ветвей

Что бы переместить результаты редактирования в побочных ветвях в мастер-ветку, необходимо выполнить слияние. Для этого выполняется переход в master и выполняется команда $git merge к побочной ветви, в результате чего все изменения в побочной ветви записываются в главную. После этого побочную ветку можно удалить, если на не нужна для дальнейшей работы параметром –d команды branch.

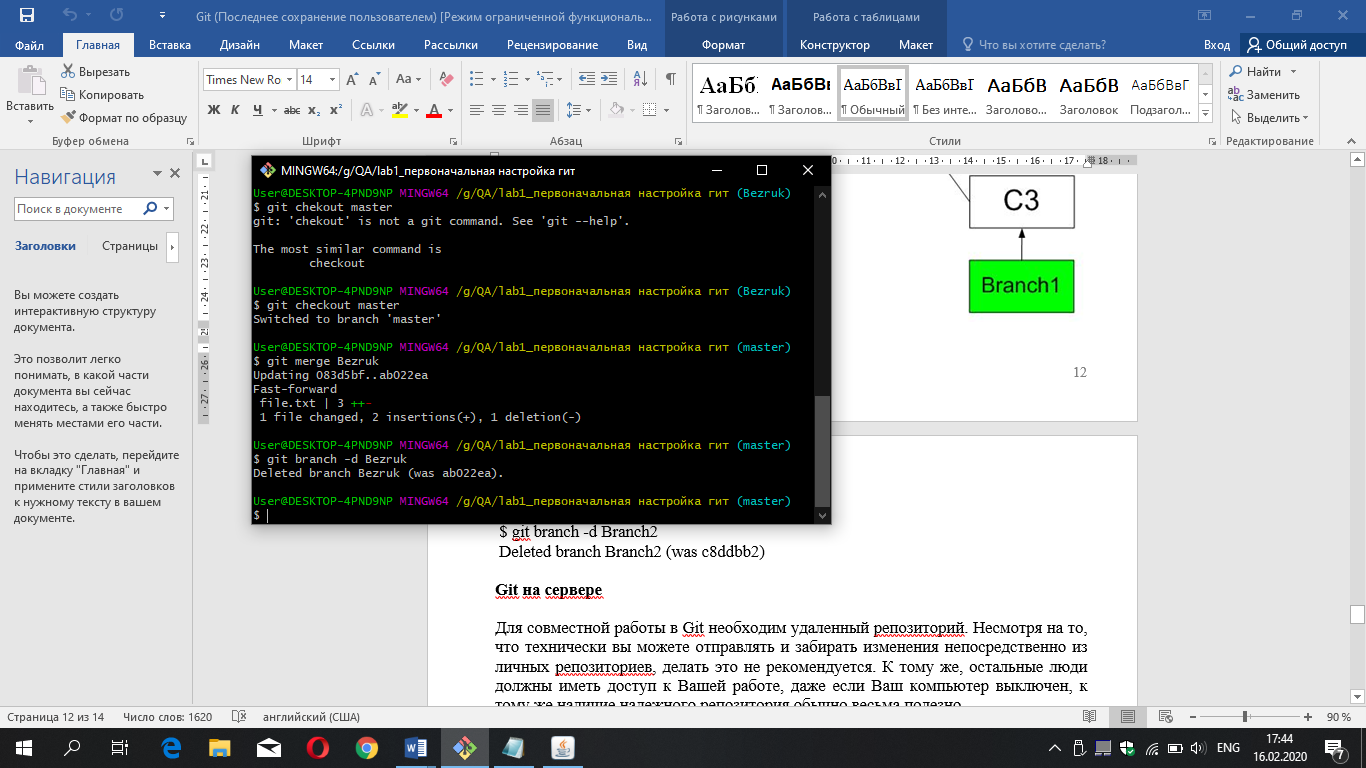


Рисунок 12 - Слияние ветвей

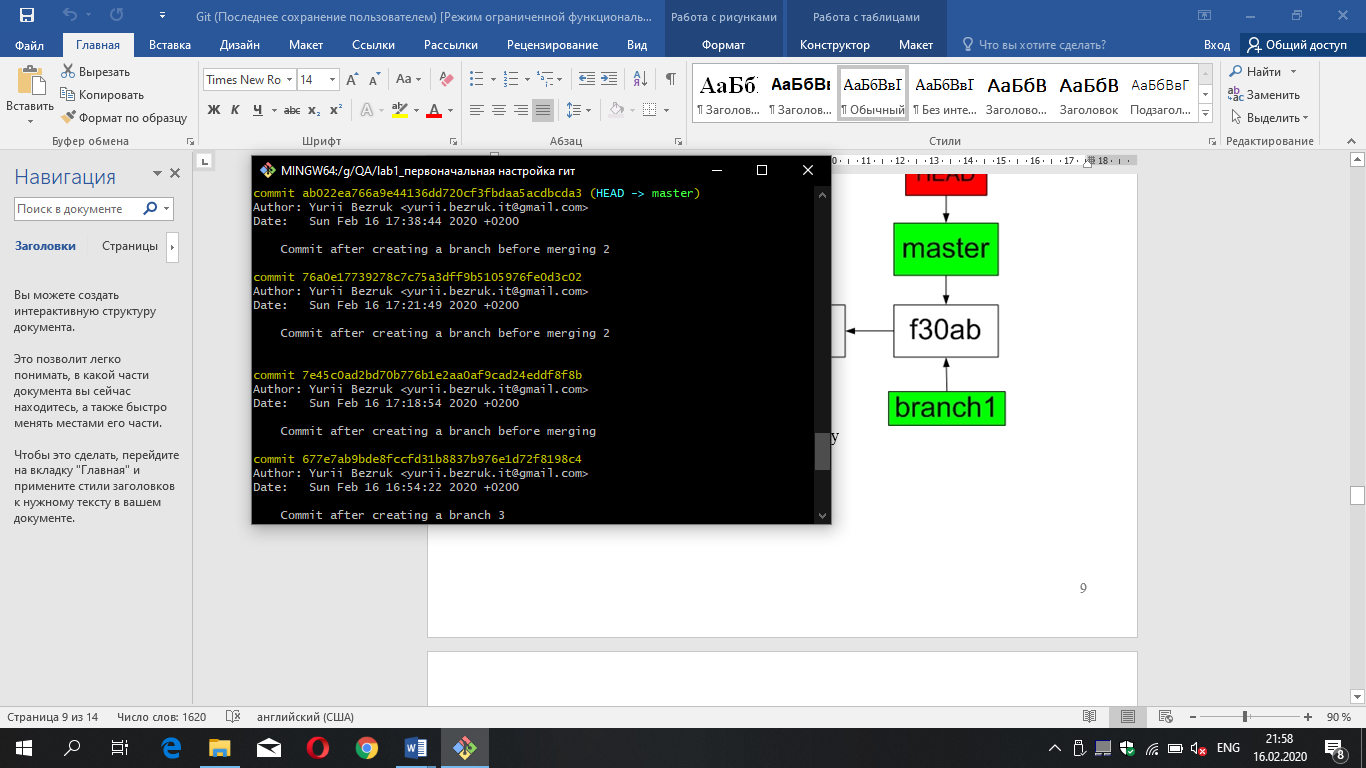


Рисунок 13 - История изменений файла

1. Регистрация на GitHub

GitHub – удобный хостинг для системы Git, позволяющий хранить данные на удаленном репозитории. После регистрации был создан профиль на странице https://github.com/

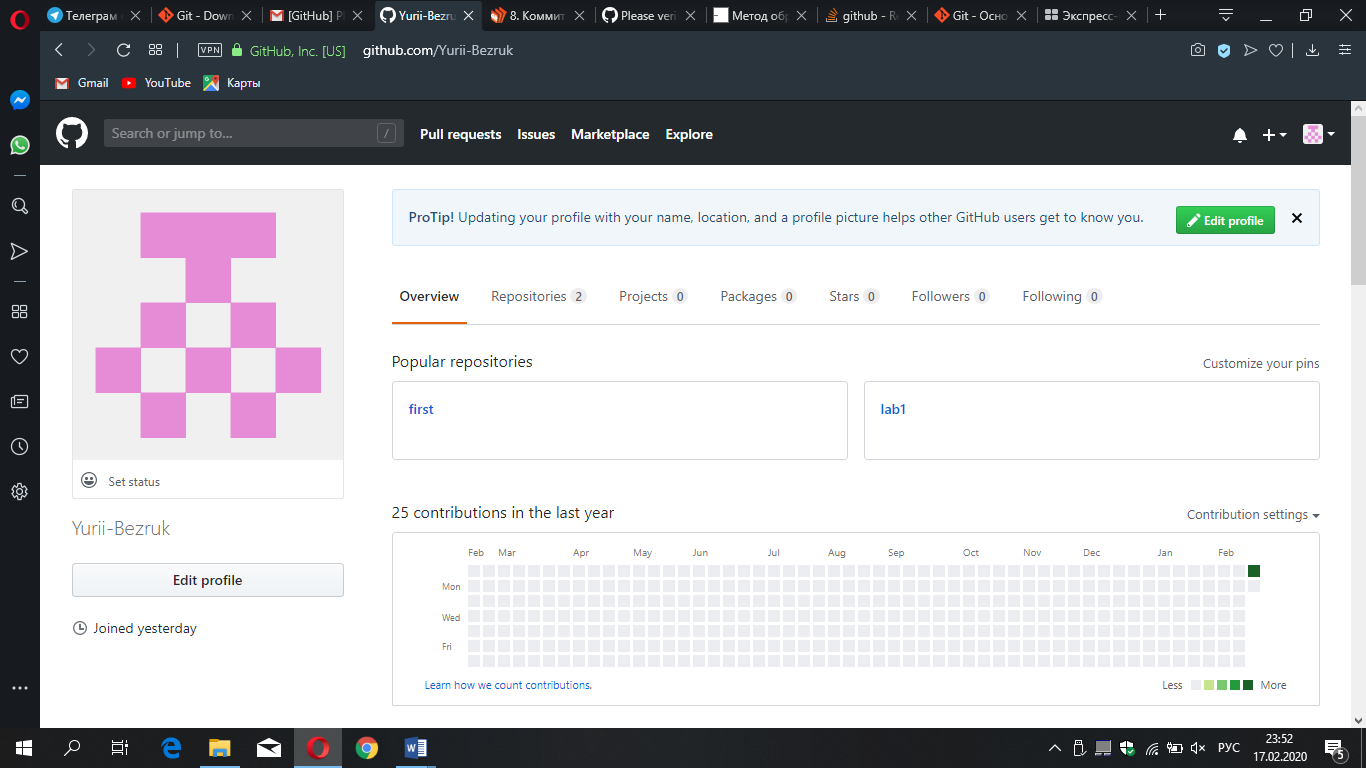


Рисунок 14 - Профиль GitHub

1. Работа с удаленным репозиторием

Для работы с удаленным репозиторием в Github необходимо добавить remote origin следующими командами:

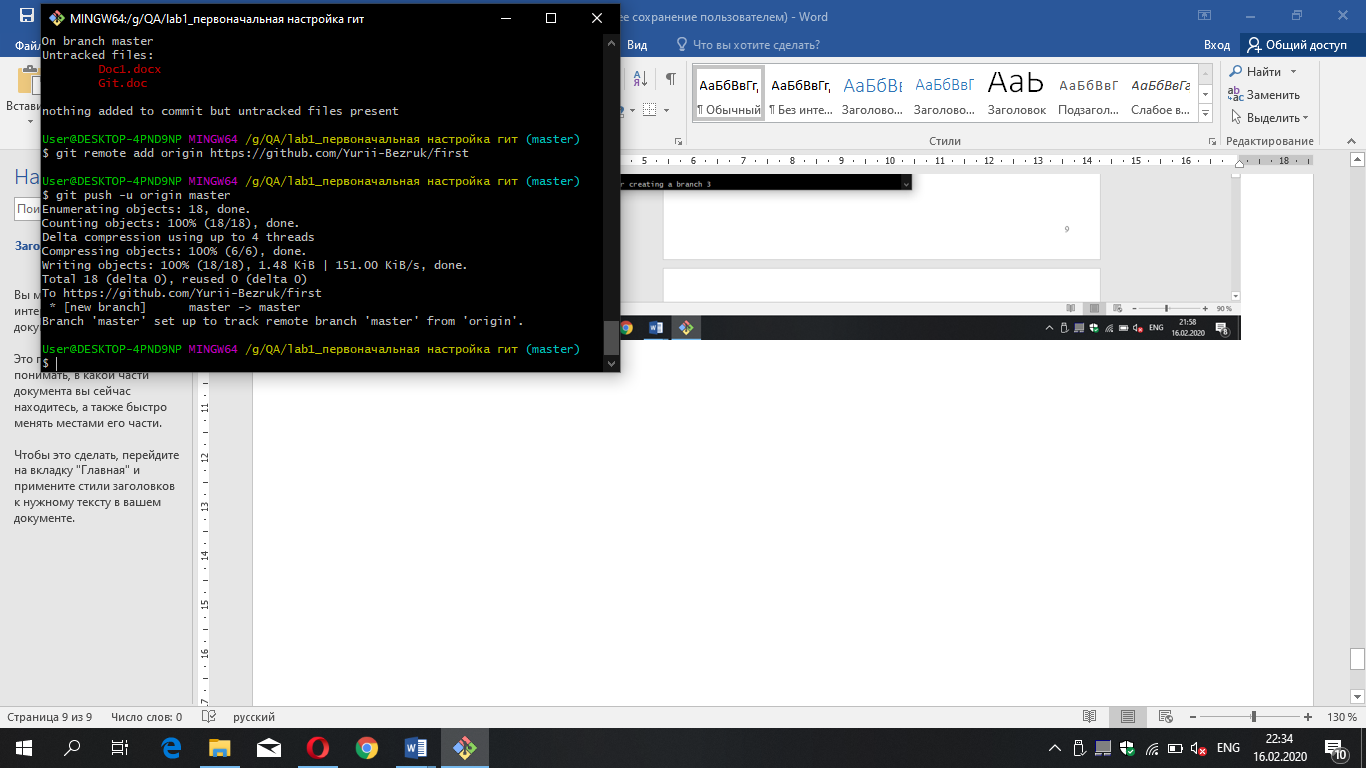


Рисунок 15 - push проекта на GitHub

После чего изменения в файле после коммита можно сохранять не только в локальной директории но и в удаленной:

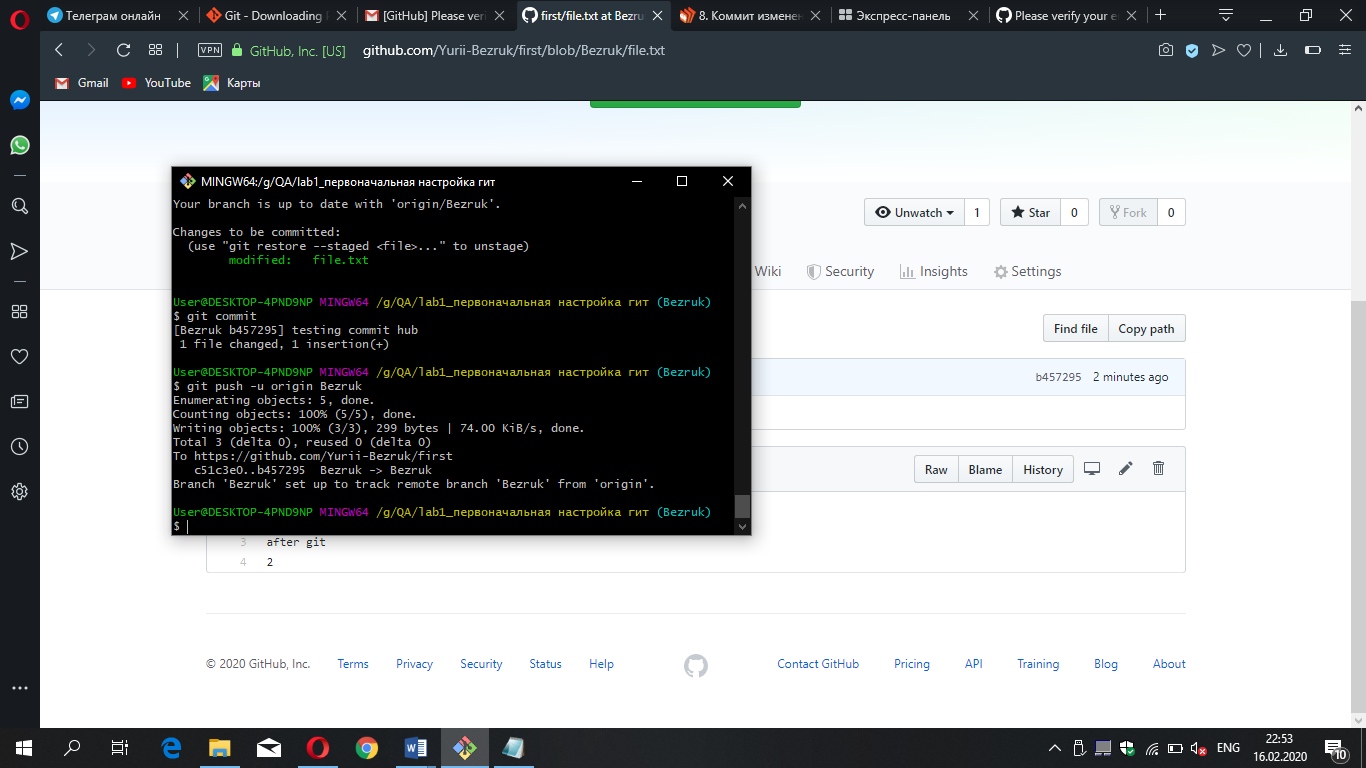


Рисунок 16 - Фиксация изменений в файлах на GitHub

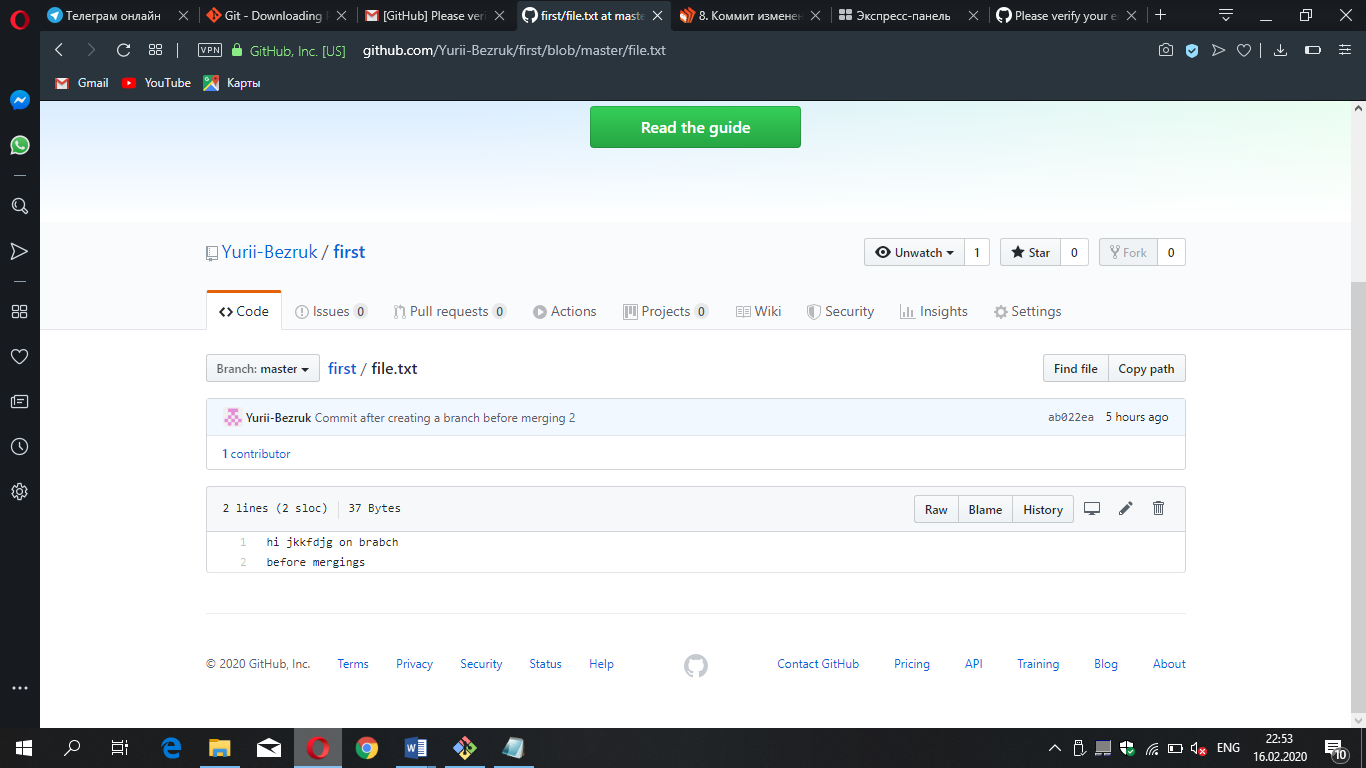
Далее приведены результаты удаленной фиксации: загружены обе ветки с разными коммитами, поэтому при переключении веток в репозитории на GitHub наблюдаются разные версии одного файла. 

Рисунок 17 - Файл в мастер-ветке

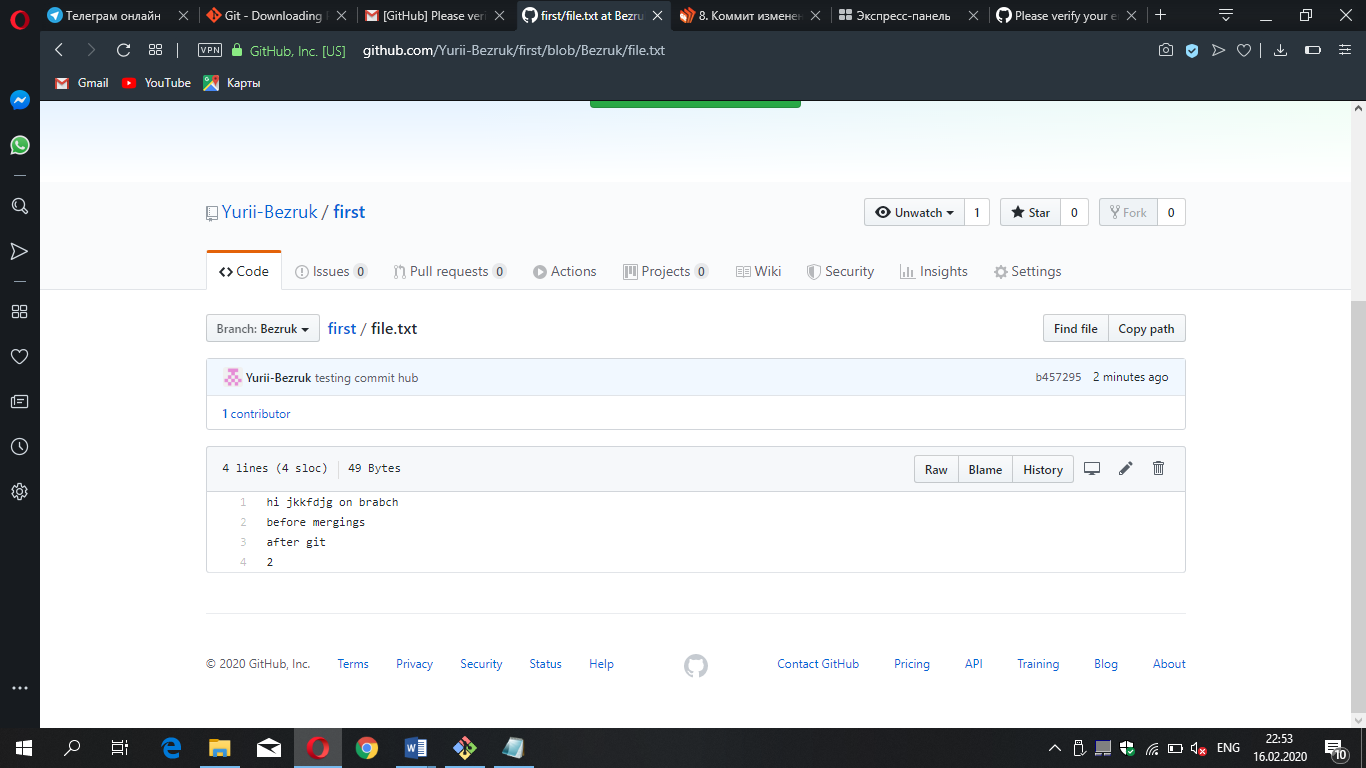


Рисунок 18 - Изменённый файл в побочной ветке

При чем изменения в ветках могут вообще не касаться одного файла. Далее выполняется Git Bash в папке проекта Java и на новый репозиторий lab1 в GitHub с разных веток заливаются первичный текстовый файл и все файлы проекта. При переключении веток наблюдается изменение набора файлов репозитория.

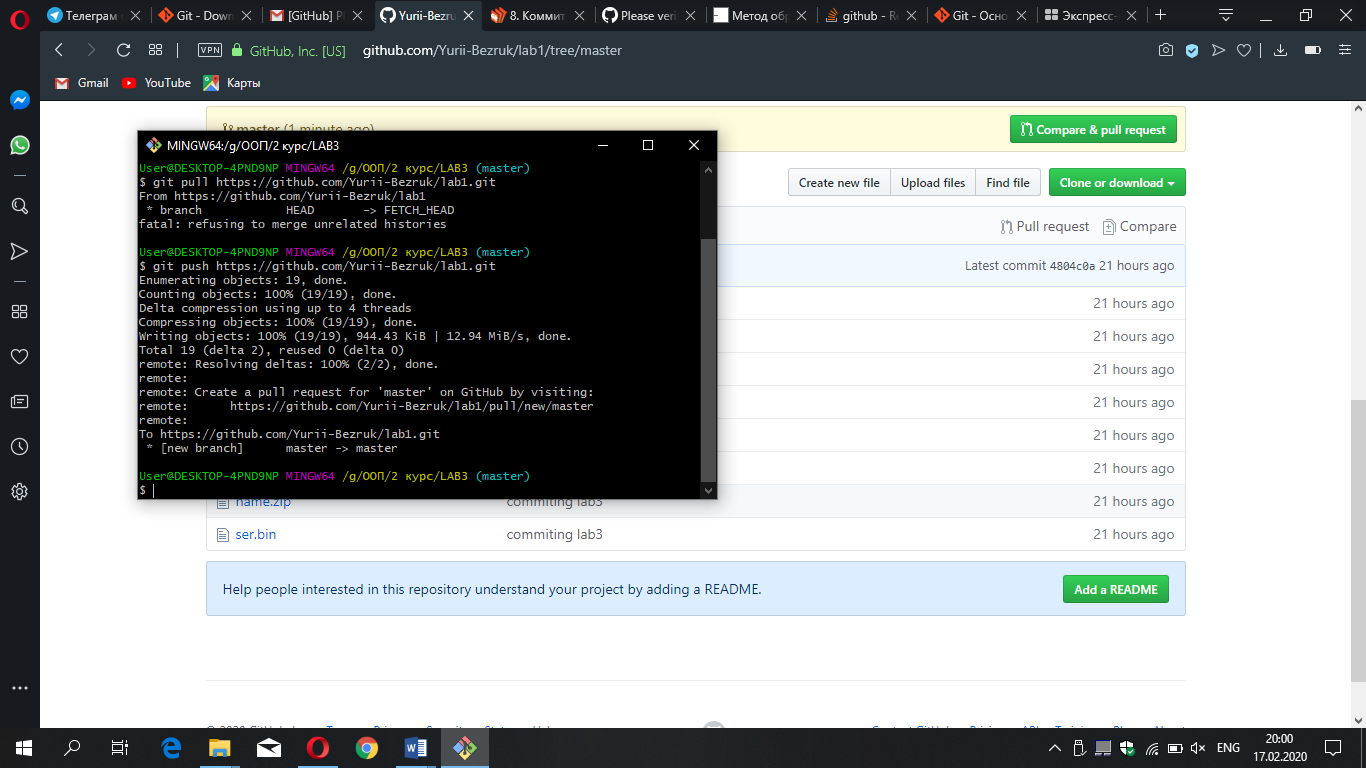


Рисунок 19 - push проекта на GitHub

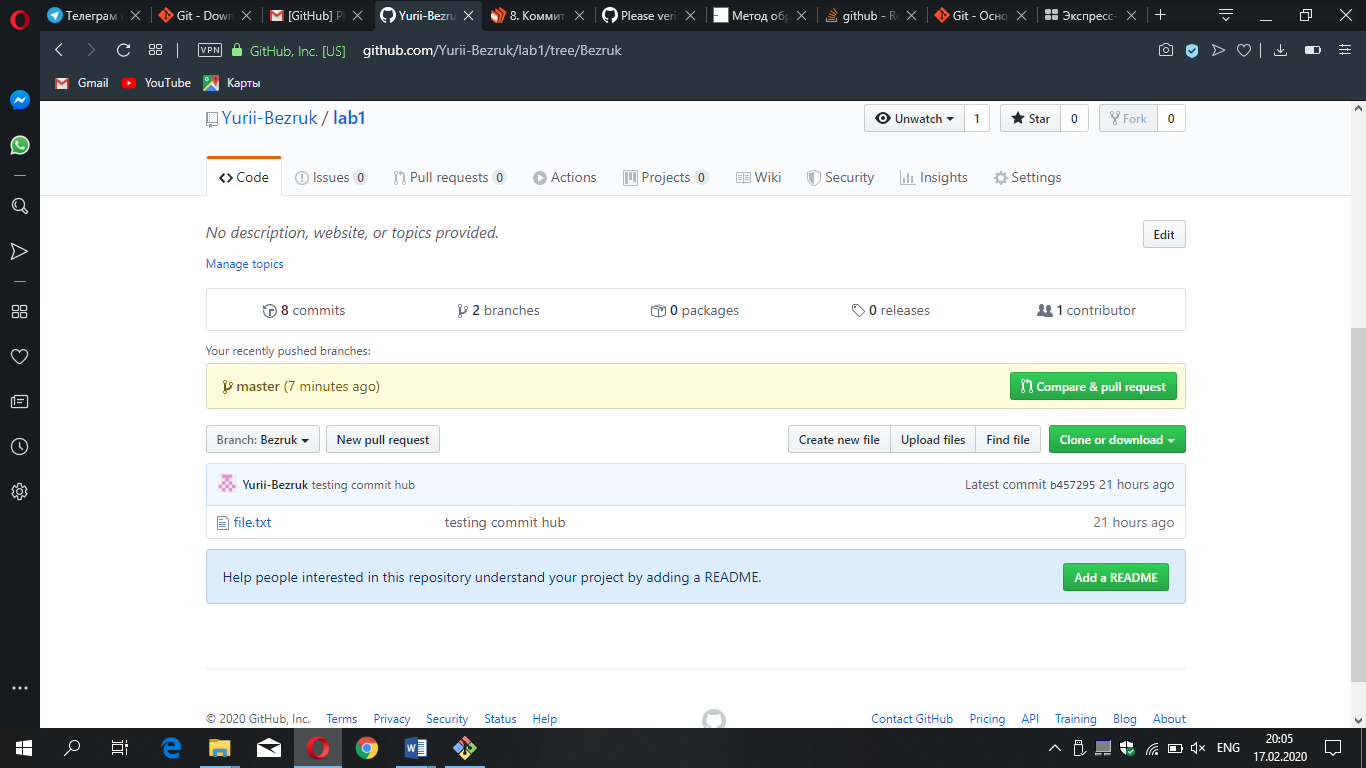


Рисунок 20 - Файлы в мастер-ветке

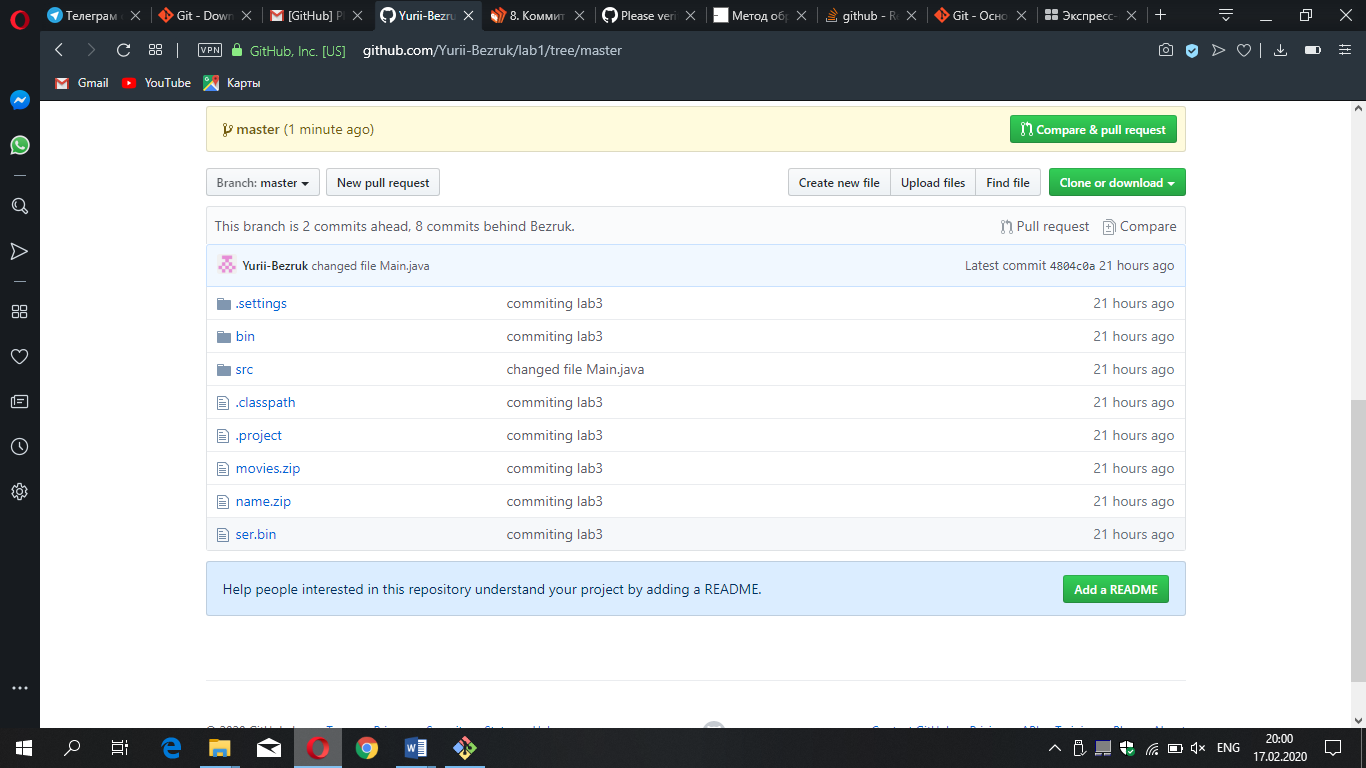


Рисунок 21 - Файлы в побочной ветке

ВЫВОДЫ

Git является удобной системой контроля версий программного продукта, позволяющей эффективно управлять разработкой проекта. При помощи системы удаленных репозиториев Git так же очень удобен для работы в команде. В ходе выполнения рабораторной работы было проведено ознакомление с основными возможностями и функциями системы и получена рабочая версия Git на компьютер. При помощи технологии ветвления и коммитов можно осуществлять эффективные переходы между версиями проекта, что очень удобно для работы.

Міністерство освіти і науки України

Харківський національний університет імені В.Н. Каразіна

Факультет комп’ютерних наук

**ЛАБОРАТОРНА РОБОТА**

з дисципліни

«Математичні методи та технології тестування та верифікації програмного забезпечення»

Тема: Вивчення системи контролю версій на прикладі Git.

Виконав студент 2 курсу

групи КС-21

Золотухін Володимир Олександрович

Харків – 2019