**Лабораторная работа №1**

**Основы *Git* и *Github***

**Цель работы:**

Выполнение лабораторной работы направлено на изучение:

1. Наиболее распространенных практик в области контроля версий программного обеспечения, его использования в командной разработке ПО и *DevOps*;

2. Концепции *Git*, основанной на понятиях репозитория и ветвления версий ПО;

3. Порядка использования *GitHub* и его базовых операций.

**Порядок работы:**

1. Зарегистрировался в *GitHub*, как показано на рисунке 1.

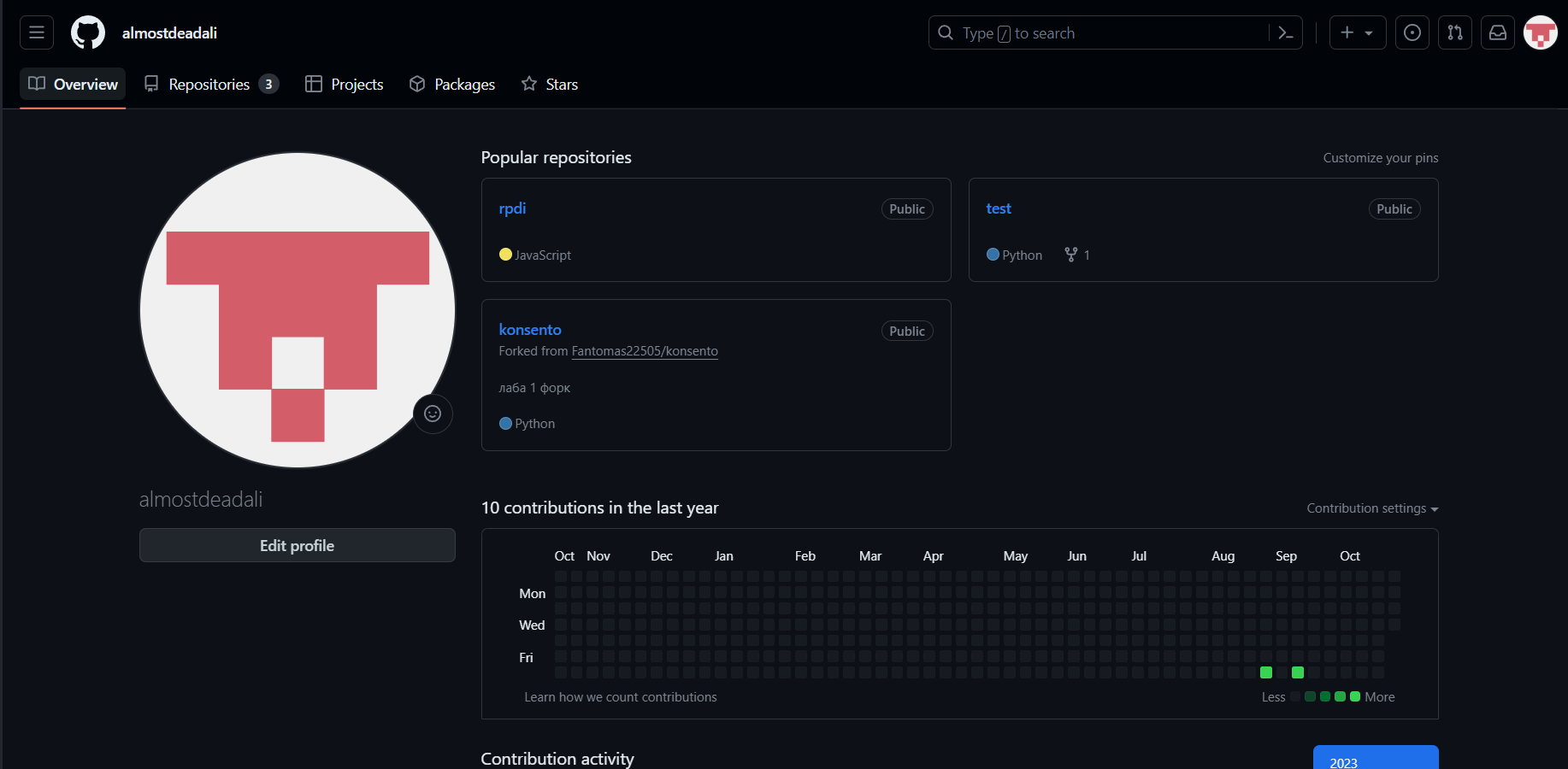


Рисунок 1 - Аккаунт GitHub

2. Создал новый репозиторий: задал имя репозитория, добавил описание, выбрал видимость репозитория «публичный», выбрал опцию «*Initialize this repository with a README*», затем отредактировал файл *README*, как показано на рисунке 2.

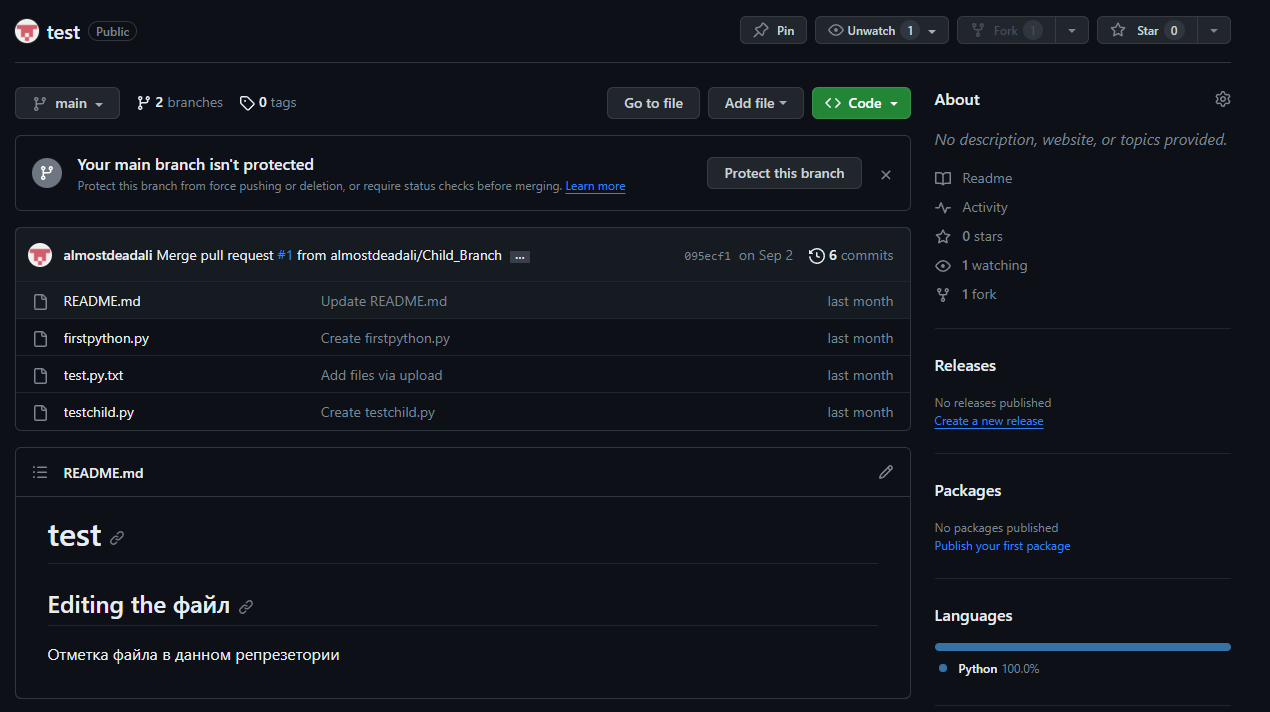


Рисунок 2 - Новый репозиторий

Создал файл *firstpython*.*py* с помощью встроенного веб-редактора *GitHub*. Зафиксировал изменения в репозитории, как показано на рисунке 3.

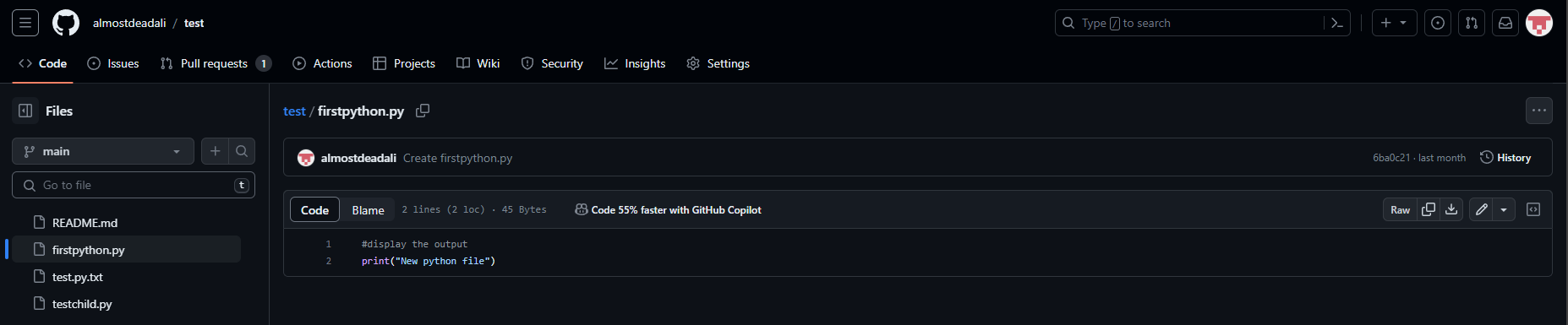


Рисунок 3 - Создание файла

3. Создал новую ветку «*Child*\_*Branch*», создал новый файл. Убедился, что файл, добавленный в дочернюю ветку, не добавляется автоматически в основную ветку, как показано на рисунке 4.

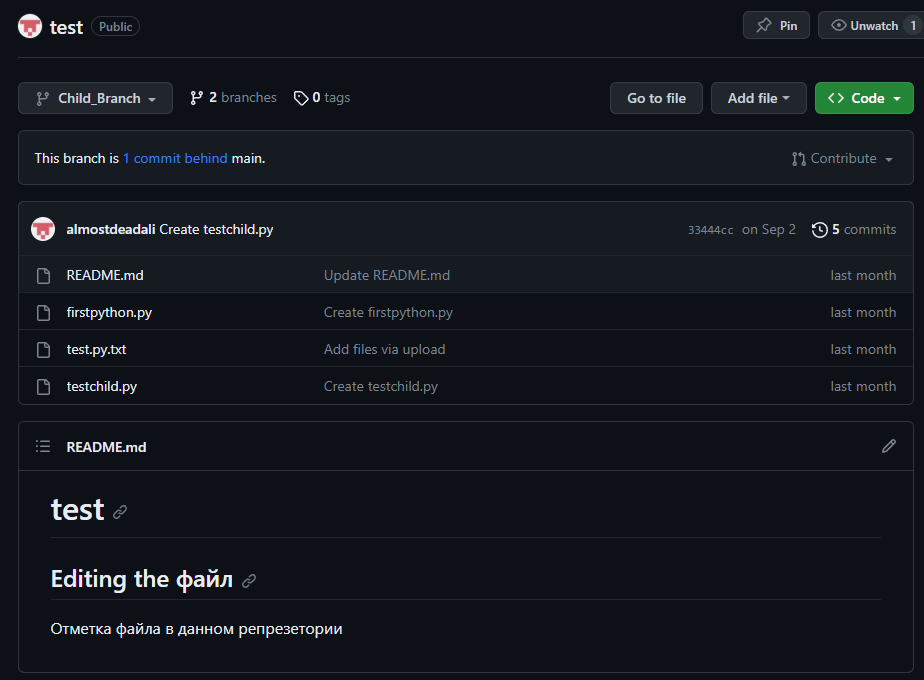


Рисунок 4 - Создание новой ветки

В *Child*\_*Branch* нажал кнопку «*Compare* & *pull* *request*», убедился, что в списке указан история ветки, как показано на рисунке 5.



Рисунок 5 - Список измененных файлов

4. Чтобы объединить ветки по запросу *pull* *request* в проекте, открыл вкладку «*Pull requests*». Отображается список ожидающих запросов на включение. Перешел на нужный *pull request* и нажал «*Merge pull request*», чтобы принять запрос на включение и объединить обновления, как показано на рисунке 6.

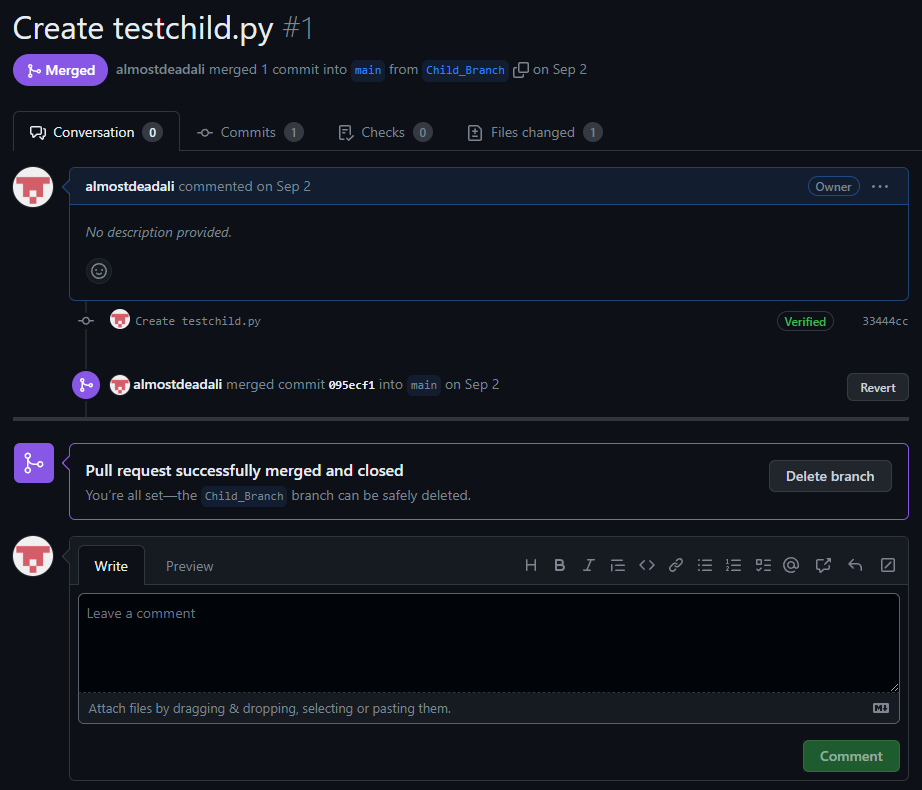


Рисунок 6 - Запрос на включение

5. Произвел работу с локальным репозиторием посредством командной строки *Windows* *PowerShell* через специальное приложение *Git*.

Создал каталог *myrepo* и перешел в созданный каталог, затем создал локальный репозиторий, вывел на экран содержимое подкаталога .*git*, в котором находится локальный репозиторий, как показано на рисунке 7.

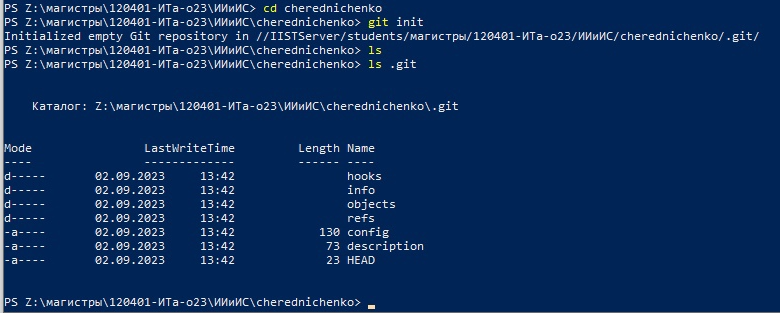
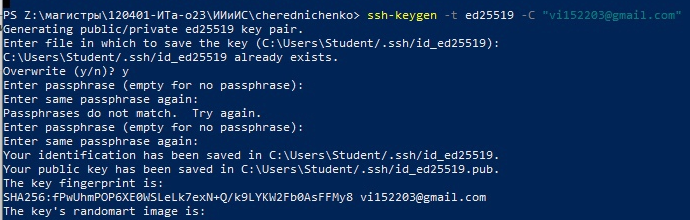


Рисунок 7 - Новый каталог

Создал пустой файл *newfile*, добавил его в репозиторий. Прежде чем зафиксировать изменения, сообщил *git* информацию пользователя. Появившийся в репозитории *newfile* зафиксировал с добавлением сообщения «*added new file*», как показано на рисунке 8.



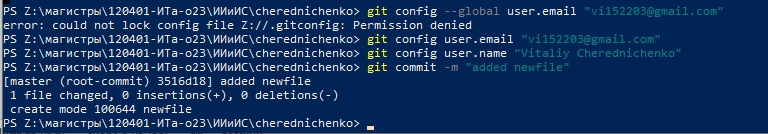


Рисунок 8 - Новый файл

Создал новую ветку под названием *my1stbranch*, убедился в наличии двух веток в репозитории и переключился с ветки *master* на ветку *my1stbranch*, как показано на рисунке 9.

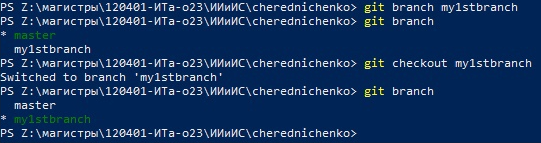


Рисунок 9 - Новая ветка

Внес изменения в *newfile* путем добавления текста, убедился что текст добавлен, как показано на рисунке 10.



Рисунок 10 - Запись и чтение из файла

Создал новый файл *readme*.*md* и добавил в репозиторий, проверил изменения в текущей ветке *my1stbranch*, затем добавил *newfile* явно, как показано на рисунке 11.

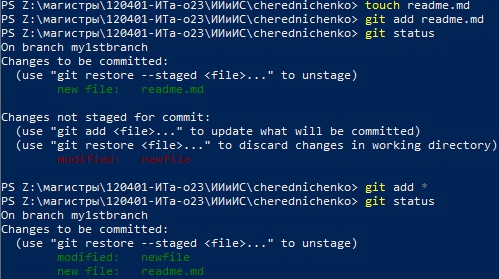


Рисунок 11 - Добавление файла

Сохранил изменения в ветку, прикрепив сообщение «*added* *readme*.*md* *modified* *newfile*», далее получил историю последних коммитов — последний коммит в *my1stbranch*, а также предыдущий коммит в *master*, как показано на рисунке 12.

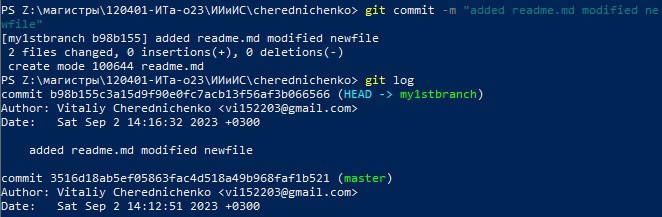


Рисунок 12 - Сохранение изменений

Произвел отмену изменений, использовав ярлык *HEAD* для отката последнего коммита, как показано на рисунке 13.

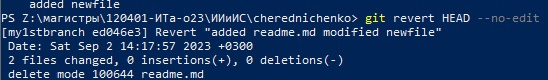


Рисунок 13 - Отмена изменений

Создал новый *goodfile* и убедился, что файл зафиксирован в *my1stbranch*, как показано на рисунке 14.

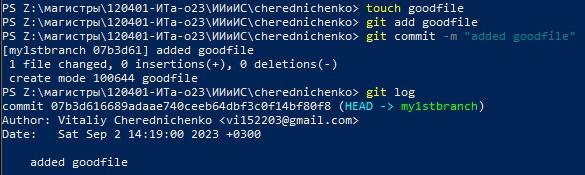


Рисунок 14 - Новый файл

Далее объединил содержимое *my1stbranch* с основной веткой, для этого сначала сделал ветку *master* активной. Произвел слияние веток и вывел на экран журнал. После того, как слияние успешно завершилось, удалил ветку *my1stbranch*, как показано на рисунке 15.

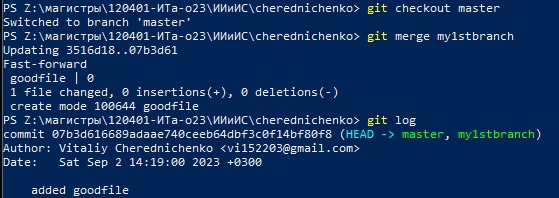


Рисунок 15 - Слияние веток

**Задание №1:**

***git* *checkout* -*b* *newbranch*** //Создал новую ветку *newbranch*

***ni* *newbranchfile*** //Создал пустой файл *newbranchfile*

***git* *add* *newbranchfile*** //Добавил файл в свою ветку

***git* *commit* -*m* "добавлен *newbranchfile*"** //Зафиксировал изменения в новой ветке

***git* *revert* *HEAD* —*no*-*edit*** //Отменил последние зафиксированные изменения

***ni* *newgoodfile*** //Создал новый файл с именем *newgoodfile*

***git* *add* *newgoodfile*** //Добавил последний файл в новую ветку

***git* *commit* -*m* "добавлен *newgoodfile*"** //Зафиксировал изменения

***git* *checkout* *master*** //Переключился на основную ветку

***git* *merge* *newbranch*** //Объединил изменения в новой ветке с основной

Результат выполнения представлен на рисунках 16-20.

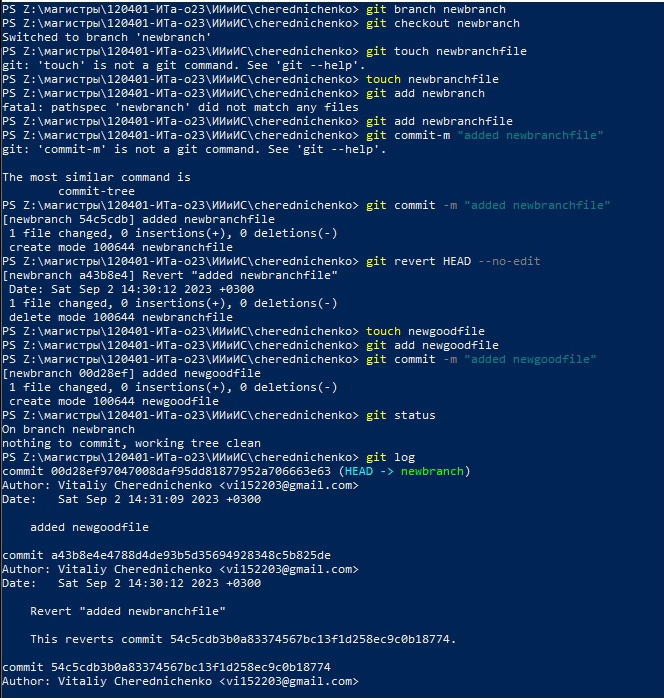


Рисунок 16 - Процесс выполнения задания

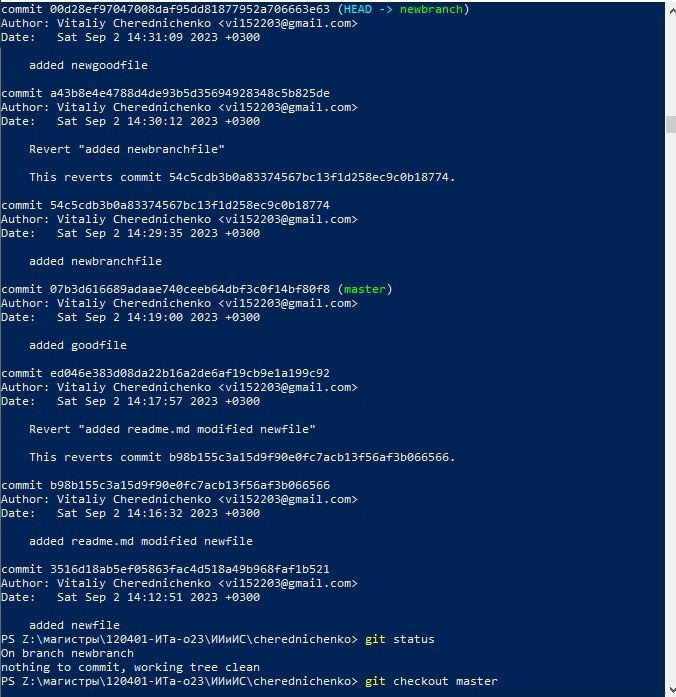


Рисунок 17 - Процесс выполнения задания

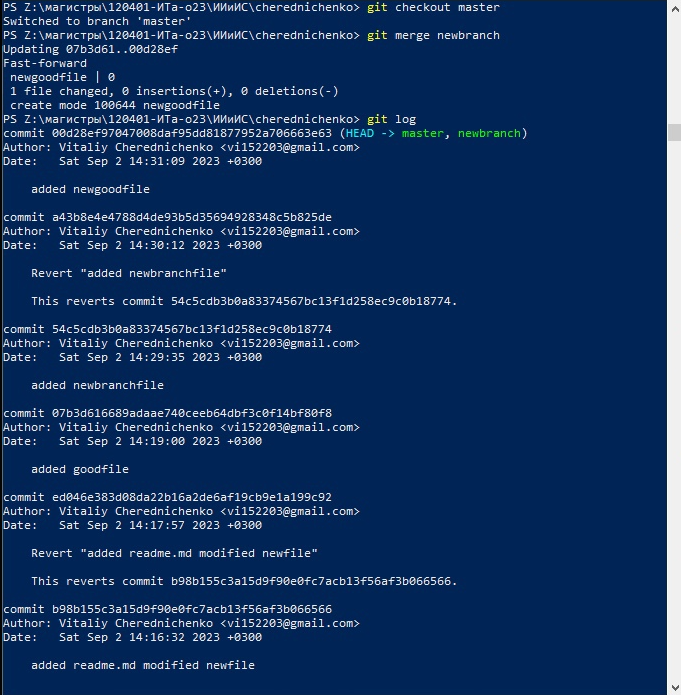


Рисунок 18 - Процесс выполнения задания

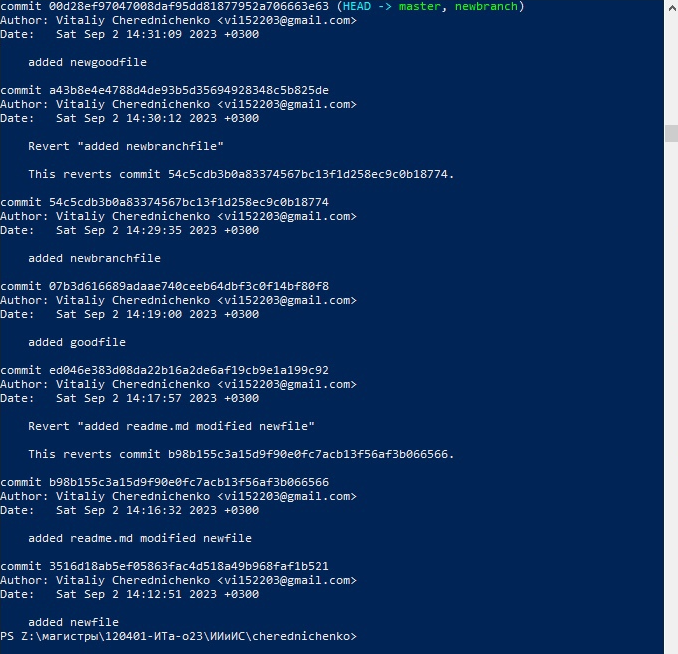


Рисунок 19 - Процесс выполнения задания

**Задание №2:**

Выполнил *fork* проекта второго студента, как показано на рисунке 20.

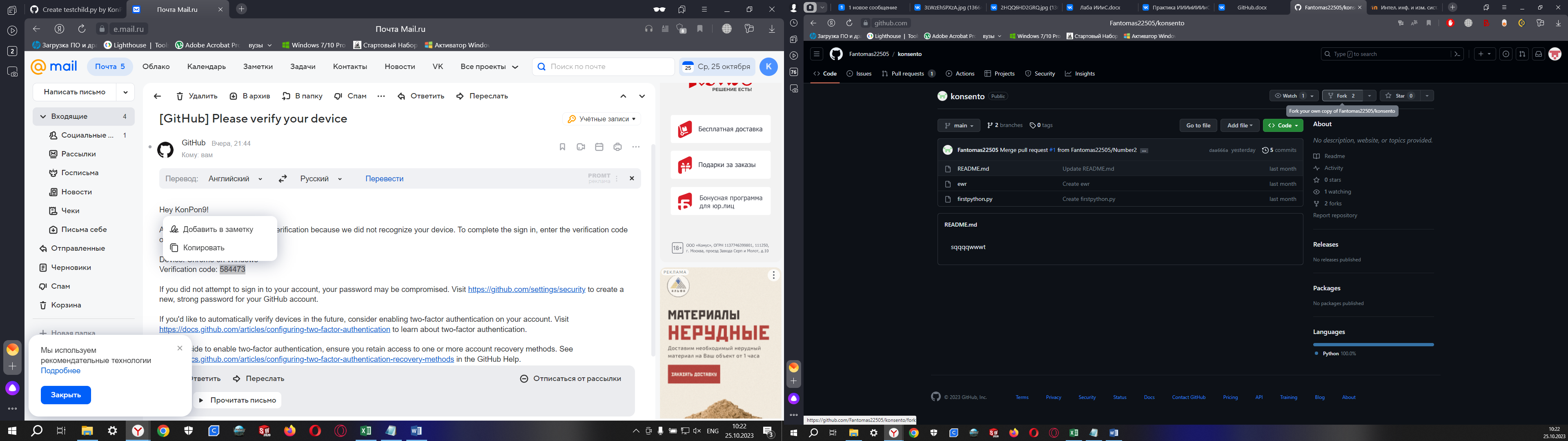


Рисунок 20 - *Fork* проекта второго студента

Выполнил *clone* проекта в локальный репозиторий с помощью *Windows* *PowerShell*, как показано на рисунке 21.

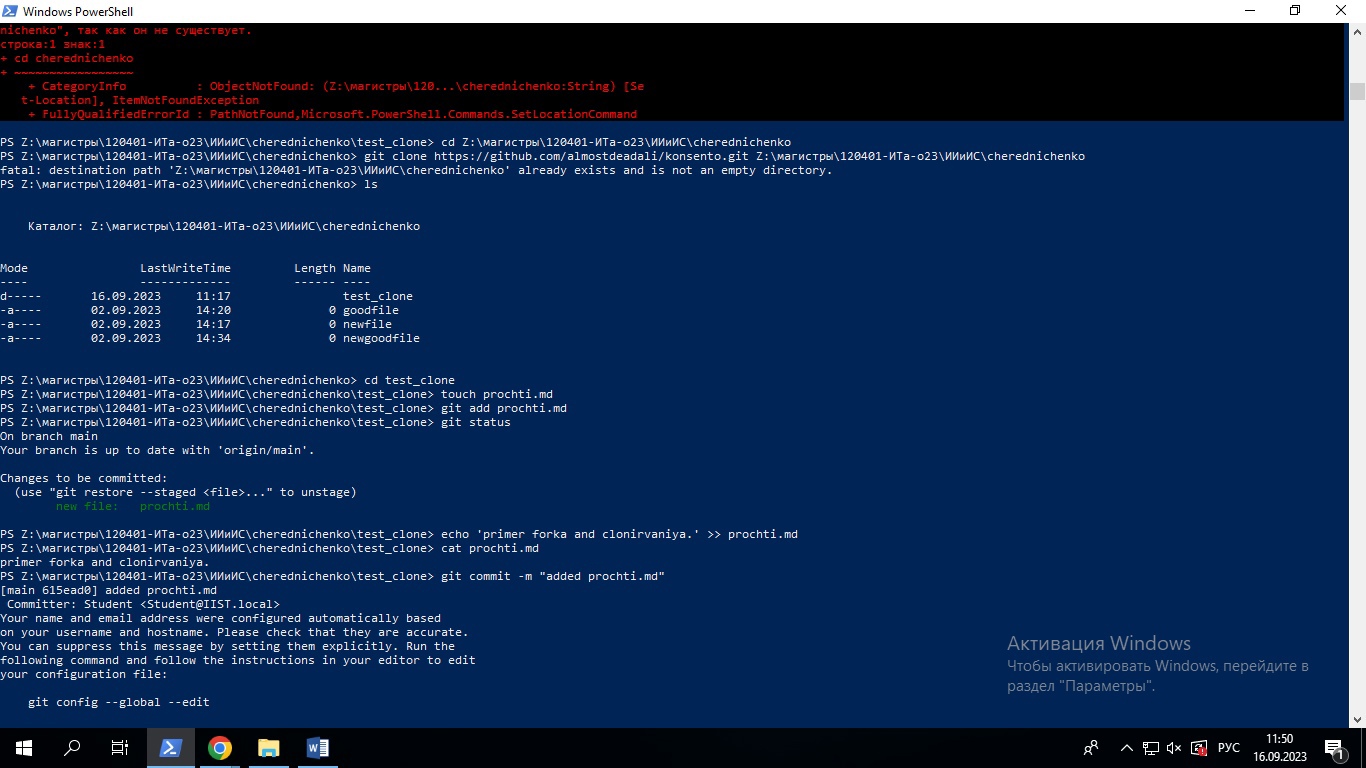


Рисунок 21 - *Clone* проекта

Добавил новый файл в репозиторий, зафиксировал изменения, как показано на рисунке 22.

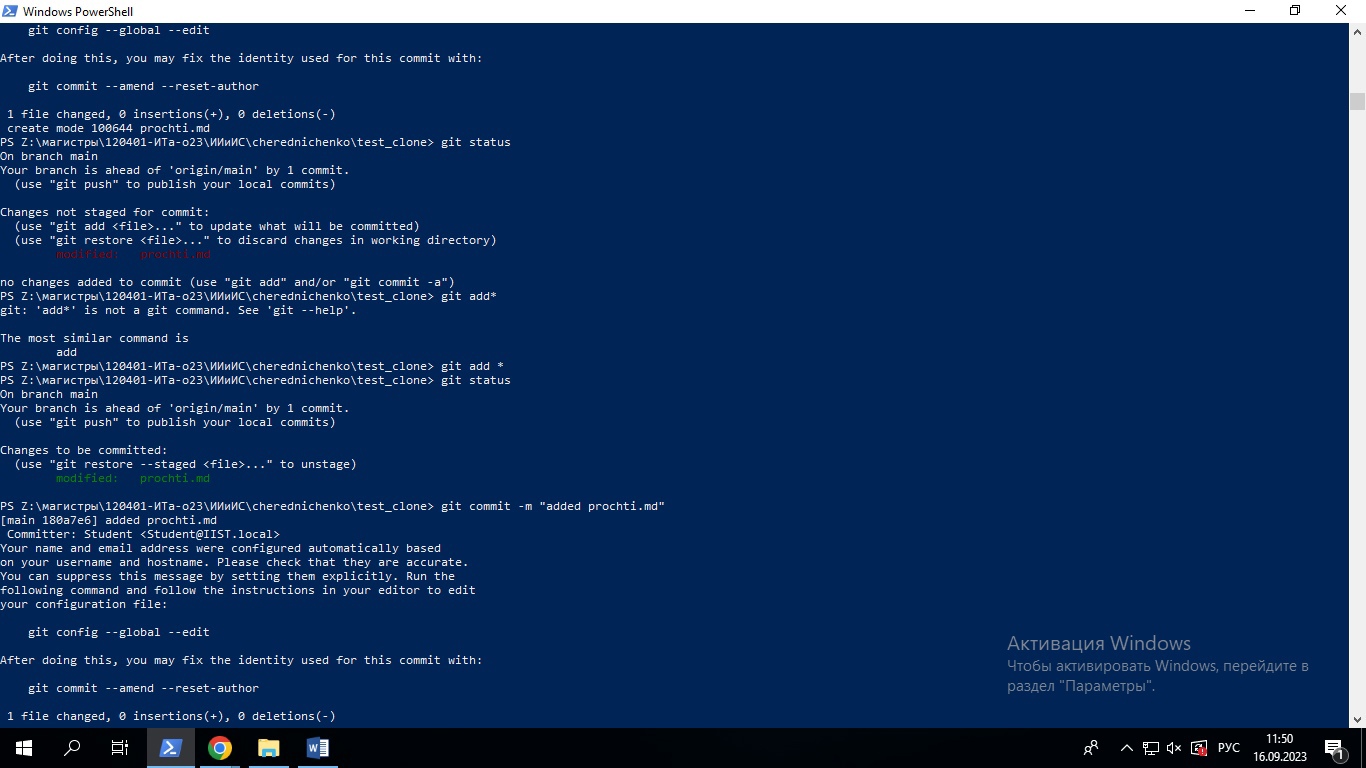


Рисунок 22 - Новый файл

Выполнил синхронизацию с *fork*-репозиторием, как показано на рисунке 23.

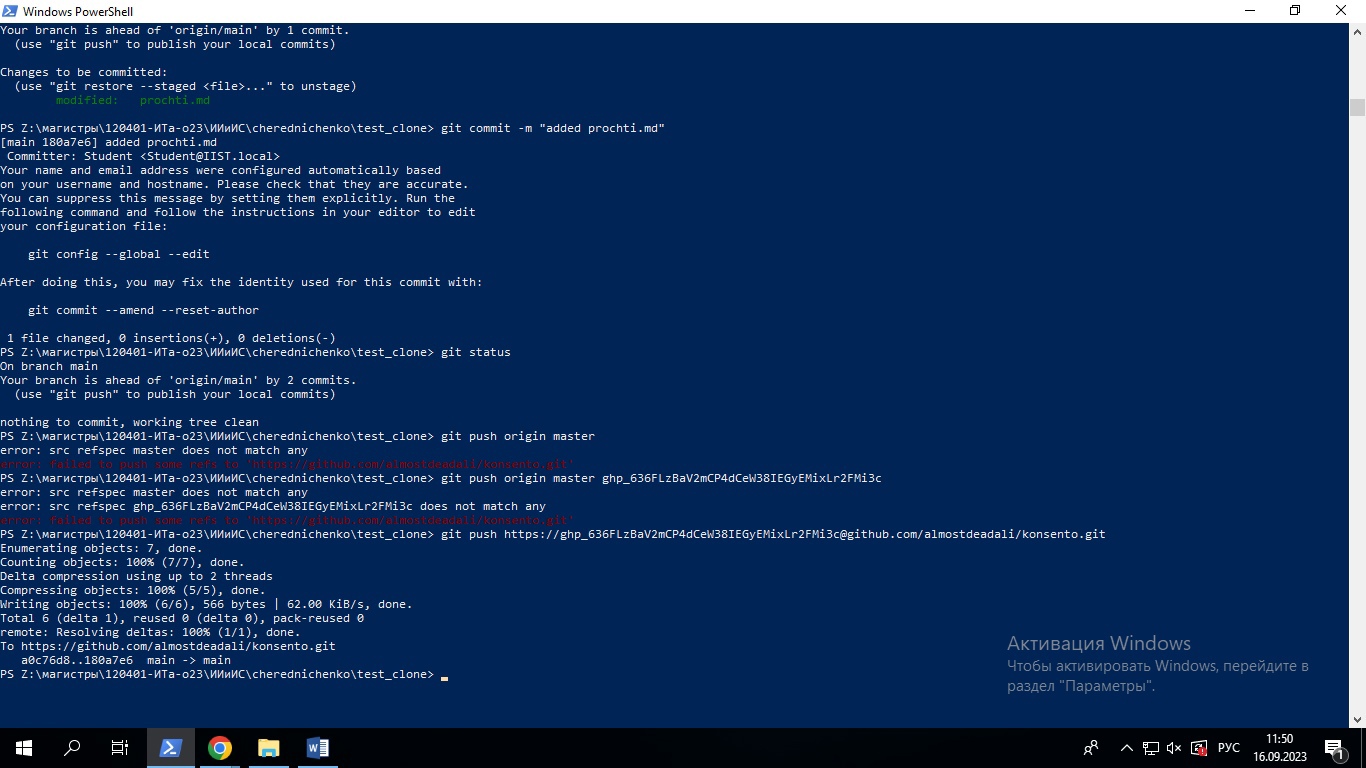


Рисунок 23 - Синхронизация с *fork*-репозиторием

Сформировал *pull* *request* к *origin* проекту на прием данного изменения, как показано на рисунке 24.

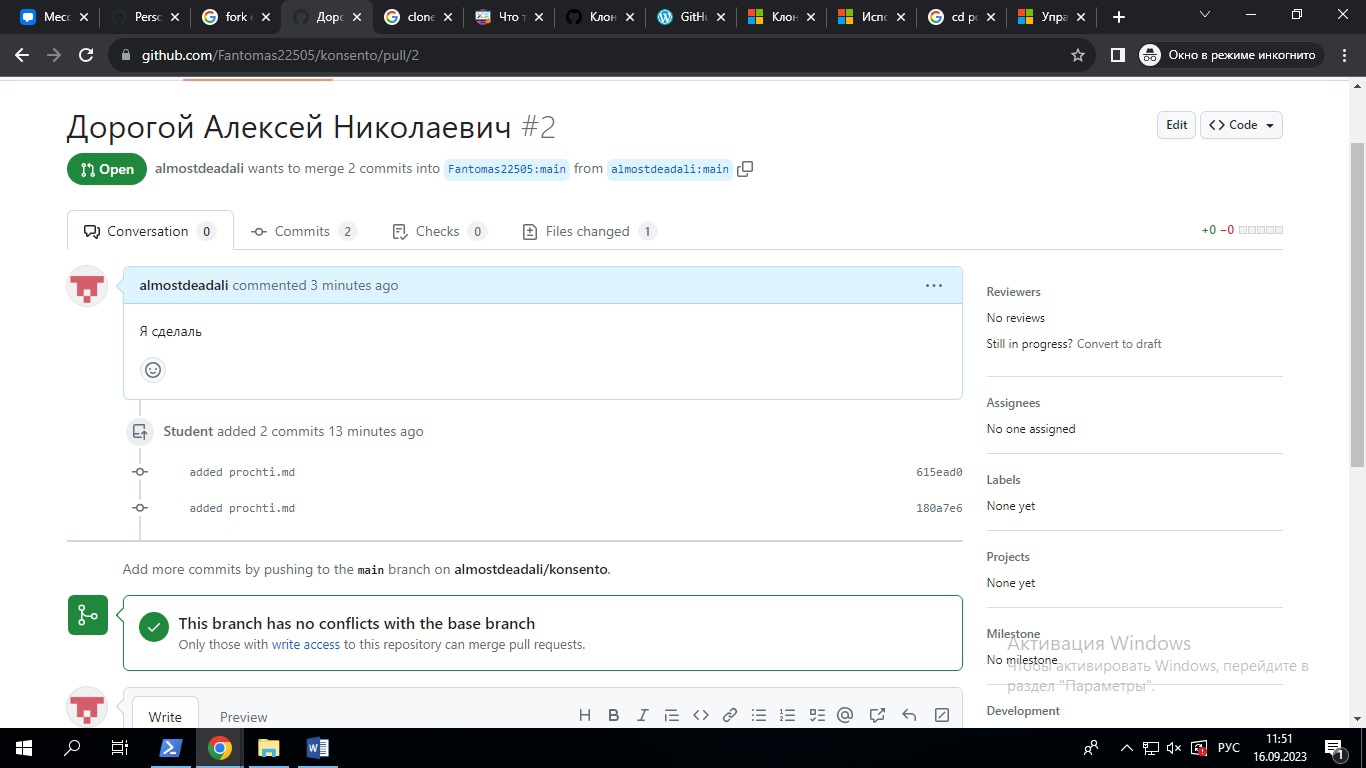


Рисунок 24 - Формирование *pull* *request*

**Вывод:** изучили концепцию *Git*, основанную на понятиях репозитория и ветвления версий ПО, изучили порядок использования *GitHub* и его базовых операций, научились работать с онлайн-хостингом *GitHub*.